MT8395 IoT Application Processor Datasheet

Version: 1.8

Release date: 2024-04-02

Version History

Version	Date	Description
1.0	2022-02-10	First official release
1.1	2022-10-28	Modified Section 1.2 Ordering Information
1.2	2022-12-09	• Modified Section 1.2 Ordering Information • Modified Section 4.4 Reserved and Unused Pin Handling Recommendations
1.3	2023-01-06	• Modified Table 5-2 Storage Conditions
1.4	2023-03-28	• Removed Table 3-65 Boot Flash Selection and added descriptions to Section 3.14 Boot Flash • Modified the typos of ball locations in Table 3-57 GPIO Signal Descriptions and of signal names in Table 4-10 Pin Characteristics • Modified the signal type and power domain of X26M_IN in Table 4-10 Pin Characteristics • Added AVDD12_EARCRX, AVDD18_EARCRX, AVDD08_HDMIRX21, DVDD18_IODGI, DVDD18_VQPS and related information to Table 5-1 Absolute Maximum Ratings and Table 5-3 Recommended Operating Conditions • Modified the descriptions of DVDD18_IOEMMC, DVDD18_MSDC1, DVDD18_MSDC2 and DVDD28_IODGI in Table 5-3 Recommended Operating Conditions • Added a feature in Section 3.10.2 Video Decoder (VDEC)
1.5	2023-09-05	• Modified the order# and temperature range in Table 1-2 Ordering Information • Enhanced Chapter 3 Features Description and Section 5.5 by providing additional details • Streamlined the presentation of Section 3.6 Memory and Section 3.8 Display for improving clarity and organization • Modified the order# and values in Table 7-1 MT8395 AV/ZA Thermal Operating Specifications and added Table 7-2 MT8395 IV/KZA Thermal Operating Specifications
1.6	2023-09-20	• Added MT8395IV/ZA to Table 1-2 Ordering Information • Modified the maximum lane speed of DSI_TX from 1.6Gbps to 1.2Gbps in Section 3.8.1.2 Features • Modified the high-speed data rate from 1500Mbps to 1200Mbps in Table 5-14 DSI D-PHY TX Electrical Characteristics • Modified the maximum speed of DSI D-PHY from 1.5Gbps to 1.2Gbps in Table 6-1 Maximum Performance Ratings • Added Table 7-2 MT8395IV/ZA Thermal Operating Specifications
1.7	2023-11-20	• Removed operating temperature from Table 3-200 Electrical Characteristics and Operating Conditions of 1.8V Applications • Modified Table 5-3 Recommended Operating Conditions • Modified the descriptions of Function code 1 and Function code 2 in Table 7-4 Printed Device Reference and Decoding
1.8	2024-04-02	• Modified “Temperature Range” to “Operating Junction Temperature” in Table 1-2 Ordering Information • Revised the typos of ball location by changing F34 to F37 and F37 to F34 in Table 3-98 TDM Signal Descriptions, Table 3-101 DMIC Signal Descriptions and Table 3-104 I2C/I3C Signal Descriptions • Added junction temperature to Table 5-1 Absolute Maximum Ratings • Modified the value of θ_{JA} in Table 7-3 MT8395IV/KZA Thermal Operating Specifications

Table of Contents

List of Figures

List of Tables

1 Introduction

The MT8395 is a highly integrated platform incorporating the following key features:

- Quad-core Arm® Cortex®-A78 processor
- Quad-core Arm Cortex-A55 processor
- Arm Mali™-G57 MC5 3D Graphics Accelerator (GPU) with Vulkan® 1.1, OpenGL ES 3.2 and OpenCL™ 2.2
- Dual-core AI Processor Unit (APU) Cadence® Tensilica®
- VP6 processor with AI Accelerator (AIA)
- Single-core Cadence HiFi 4 Audio Engine DSP
- LPDDR4X: Up to 16GB, with memory data rate up to LPDDR4X-4266
- Display output supporting 4K60 + 4K60 resolution
- Image processing: 48MP @ 30fps for single camera capture; 16MP + 16MP @ 30fps for dual camera capture
- Video encoding: 4K @ 60 fps with HEVC/H.264
- Video decoding: 4K @ 90 fps with AV1/VP9/HEVC/H.264

Figure 1-1 shows the functional block diagram of the device.

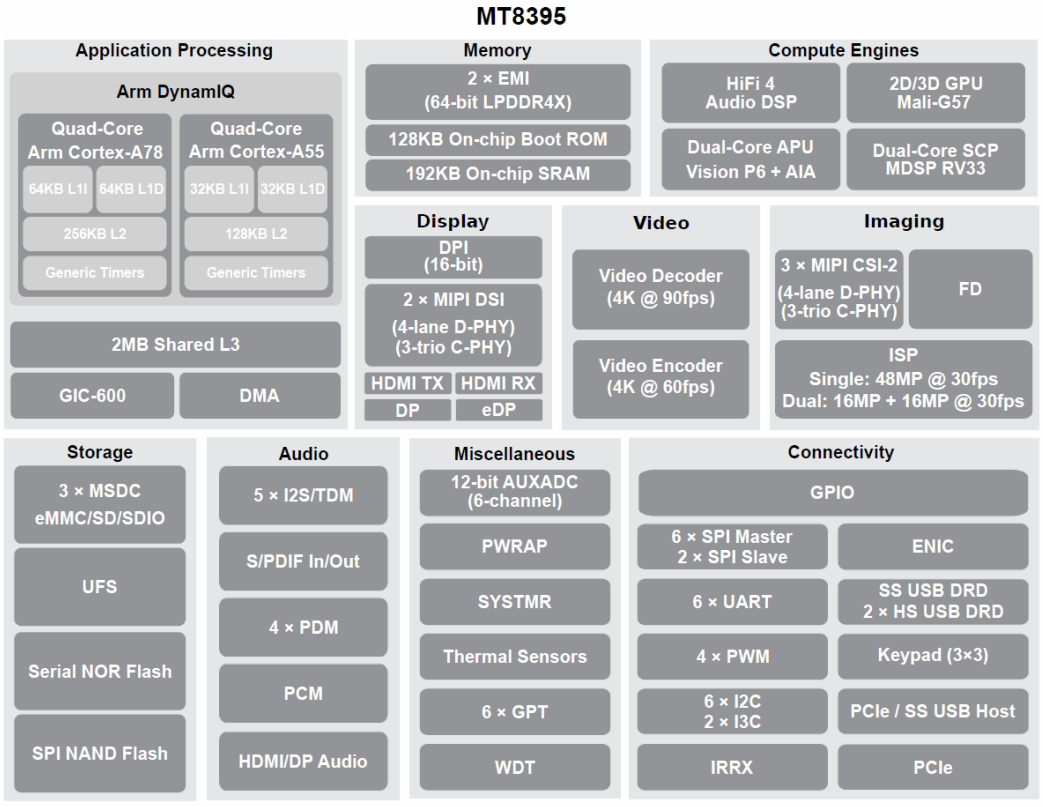


Figure 1-1 Functional Block Diagram

NOTE: Hardware components may work differently on different operating systems.

1.1 Features Overview

Table 1-1 presents a summary of the device feature.

Table 1-1 Device Features

Feature		MT8395
Processors		
Quad-core Arm Cortex-A78	A78	2200 MHz
Quad-core Arm Cortex-A55	A55	2000 MHz
Graphics Accelerator Mali G-57 MC5	GPU	880 MHz
HiFi 4 Digital Signal Processor	DSP	720 MHz
AI Processor Unit	APU	832 MHz
System Companion Processor	SCP	416 MHz
Memory		
External Memory Interface (LPDDR4X)	EMI	Up to 16GB LPDDR4X-4266
Storage		
Memory Card Controller eMMC™/SD®/SDIO	MSDC0	eMMC (1-/4-/8-bit)
	MSDC1	SD Card (1-/4-bit)
	MSDC2	SD/SDIO Card (1-/4-bit)
SPI NAND Flash Interface	SNFI	Yes
Serial NOR Flash Interface	SNOR	Yes
Universal Flash Storage	UFS	Yes (1-lane)
Display		
High-Definition Multimedia Interface Transmitter	HDMITX	Yes (HDMI™ 2.0b)
Digital Display Parallel Interface	DPI	16-bit
DisplayPort Interface	DPTX	Yes (DP 1.4)
Embedded DisplayPort Interface	EDPTX	Yes (eDP 1.4)
MIPI™ Display Serial Interface	DSI0	4-lane D-PHY, or 3-trio C-PHY
	DSI1	4-lane D-PHY, or 3-trio C-PHY
High-Definition Multimedia Interface Receiver	HDMIRX	Yes (HDMI™ 2.0b)
Imaging		
Image Signal Processor	ISP	Single camera: 48MP @ 30fps
		Dual camera: 16MP + 16MP @ 30fps
MIPI Camera Serial Interface 2	CSI0	2 × 4-lane D-PHY
	CSI1	1 × 4-lane D-PHY, or 2 × 2-lane D-PHY, or 1 × 3-trio C-PHY, or 2 × 2-trio C-PHY
Face Detection	FD	Yes
Warp Engine	WPE	Yes
JPEG Encoder	JPEG	Baseline encoding and decoding (250 MP/s)
Video		
Video Encoder	VENC	HEVC/H.264, 4K @ 60 fps
Video Decoder	VDEC	AV1/VP9/HEVC/H.264, 4K @ 90 fps
Audio		
Inter-IC Sound	I2S	4 (2 input, 2 output)
Time Division Multiplexed Interface	TDM	
Pulse Code Modulation	PCM	1
Pulse Density Modulation (Decoder for DMIC)	PDM	4 × stereo
	SPDIF_IN	2

Feature		MT8395
Digital Interface	SPDIF_OUT	1
Connectivity		
Inter-Integrated Circuit	I2C	6
	I3C	2[(1)]
Universal Asynchronous Receiver/Transmitter	UART	6
Infrared Receiver	IRRX	1
Serial Peripheral Interface	SPI	6 (master mode only)
		2 (slave mode only)
Universal Serial Bus	USB Port 0	SS USB 3.1 Gen1 DRD
	USB Port 1	SS USB 3.1 Gen1 Host[(2)]
	USB Port 2	USB 2.0 DRD
	USB Port 3	USB 2.0 DRD
KeyPad Scanner	KeyPad	3 × 3
General Purpose I/O pins	GPIO	144
Pulse Width Modulation	PWM	Up to 4
Peripheral Component Interconnect Express	PCIe Port 0	Gen3, 2-lane, RC/EP mode
	PCIe Port 1	Gen2, 1-lane, RC mode[(2)]
Gigabit Ethernet Network Interface Controller	ENIC	MII/RMII/RGMII
Miscellaneous		
PMIC Interface	PWRAP	Yes
Auxiliary ADC	AUXADC	12-bit, 6-channel
Timers	GPT	5 × 32-bit and 1 × 64-bit
	SYSTMR	64-bit
	WDT[(3)]	Yes
Thermal Controller	TCSYS	Yes

(1) I3C0 and I3C1 support MIPI I3C® (SDR mode only).

(2) USB Port 1 shares pins with PCIe Port 1.

(3) The Watchdog Timer (WDT) is part of the Top Reset Generation Unit (TOPRGU).

1.2 Ordering Information

Table 1-2 presents the available ordering part numbers.

Table 1-2 Ordering Information

Order#	Marking	Operating Junction Temperature	PCB Thickness (T)	HDMI (RX/TX)	Package
MT8395AV/ZA	See Section 7.2 Top Marking	-20~95°C T_J	T ≤ 1.0mm	Yes	MFC VFBGA
MT8395IV/ZA	See Section 7.2 Top Marking	-40~105°C T_J	T ≤ 1.6mm	Yes	MFC VFBGA
MT8395IV/KZA	See Section 7.2 Top Marking	-40~105°C T_J	T ≤ 1.6mm	Yes	MFC VFBGA

2 Preface

2.1 Pin Characteristics and Signal Descriptions Conventions

Table 2-1 describes the column headers in all Pin Characteristic and Signal Description tables in Section 4.2 Pin Characteristics and Chapter 3 Features Description.

Table 2-1 Column Headers Description

Column Name	Explanations
Ball Name	Logical name of the ball. Note that there may exist a selection of several signals for the same ball (aux mode).
Ball Location	Ball's physical location on the chip package
Signal Name	The name of the signal for the given aux mode
Type	Pin type when configured for the given aux mode: • AI: Analog input • AO: Analog output • AIO: Analog bi-directional pin • DI: Digital input • DO: Digital output • DIO: Digital bi-directional pin • P: Power • G: Ground
Description	Description of the signal
Aux. Function	Auxiliary function mode number: • 0 through 7 are possible alternative functions • An empty box means Not Applicable and the ball is dedicated to one function only
Reset State	Shows the Aux. function configured at the release of the SYSRSTB signal
Buffer Type	Describes the associated input/output buffer type
Power Domain	Indicates the voltage supply that powers the terminal IO buffers
PU/PD	Indicates the state of an internal pull-up or pull-down resistor at the release of the SYSRSTB signal: • OFF: Internal pull-up and pull-down are disabled • PU: Pull-up is enabled • PD: Pull-down is enabled • No: Pull-up and pull-down not available • Blank cell means "No"
IO Reset Value	Shows the IO state at the release of the SYSRSTB signal

2.2 Timing Conventions, Parameters, and Information

This section provides a general description of used symbols, adopted standards and terminology, and test process. All timing characteristics are valid over the represented operating conditions unless otherwise specified.

The interface clock frequency documented in this datasheet is the maximum clock frequency, which corresponds to the maximum programmable frequency on the particular output clock. The frequency defines the maximum limit supported by the device and does not consider into account any system limitation (layouts, connectors, and so forth).

The system designer should take into account these system considerations and the device timing characteristics as well and should determine properly the maximum frequency supported to transfer the data on the corresponding interface.
The timing parameter values do not include delays by board routes. Timing values may be adjusted by increasing/decreasing such delays. If needed, external logic hardware such as buffers may be used to compensate any timing differences.

2.2.1 Timing Parameters and Information

Table 2-2 represents timing parameter symbols and descriptions used in the timing characteristic tables.

Table 2-2 Timing Parameters

Symbol	Description
f_{op}	Operating frequency
t_p	Period (cycle time)
t_d	Delay time
t_{dis}	Disable time
t_{en}	Enable time
t_h	Hold time
t_{su}	Setup time
Start	Start bit
t_t	Transition time
t_v	Valid time
t_w	Pulse duration
t_{FALL}	Fall time
t_{RISE}	Rise time
V_{OH}	High level output voltage
V_{OL}	Low level output voltage
V_{IH}	High level input voltage
V_{IL}	Low level input voltage
V_{REF}	Reference voltage

2.2.2 Parameter Information

This datasheet provides timing values at the device pin. For output timing analysis, the tester pin electronics and its transmission line effects must be also taken into account.

All timing requirements and switching characteristics are valid over the recommended operating conditions unless otherwise specified.

All rise and fall transition timing parameters are referenced correspondingly to 90% and 10% of the signal logical levels, unless otherwise specified.

2.3 Abbreviations

A

ACP
Accelerate Coherency Port
AE
Auto Exposure
AER
Advanced Error Reporting
AES
Advanced Encryption Standard
AF
Auto Focus
AFBC
Arm Frame Buffer Compression
AI
Artificial Intelligence
AIA
AI Accelerators
ALLM
Auto Low Latency Mode
APB
Advanced Peripheral Bus
APHY
Analog PHY
API
Application Programming Interface
APMCU
Application Processing Microcontroller Unit
APU
AI Processor Unit
APXGPT
Application Processor X General Purpose Timer
ASPM
Active State Power Management
ASSR
Alternative Scrambler Seed Reset
ASTC
Adaptive Scalable Texture Compression
AUXADC
Auxiliary Analog/Digital Converter
AVB
Audio Video Bridge
AWB
Auto White Balance

AXI

Advanced eXtensible Interface

B

BCLK

Baud Clock

BBC

British Broadcasting Corporation

BD

Buffer Descriptor

BDP

Buffer Descriptor Present

bps

Bits Per Second

BTA

Bus Turnaround

C

CAMSYS

Camera Imaging Subsystem

CBS

Credit-Based Shaper

CCC

Common Command Code

CCU

Camera Control Unit

CDR

Clock and Data Recovery

CEC

Consumer Electronics Control

CIC

Configuration Information Capability

CKSQ

Clock Squarer

CLK

Clock

CMDE

Command Engine

CMDQ

Command Queue

CPHA

Clock Phase

CPOL

Clock Polarity

CPU

Central Processor Unit

CPUBIU

CPU Bus Interface Unit

CRC

Cyclic Redundancy Check

CS

Complete Spilt

CSC

Color Space Conversion

CSI

Camera Serial Interface

CTS

Clear to Send

CTSI

Clear to Send Interrupt

CSMA/CD

Carrier Sense Multiple Access with Collision Detection

CV

Computer Vision

D

DA

Destination Address

DAA

Dynamic Address Assignment

DCM

Dynamic Clock Management

DDC

Display Data Channel

DE

Data Enable

DFS

Dynamic Frequency Scaling

DLL

Divisor Latch LS (Least Significant Bit)

DLM

Divisor Latch MS (Most Significant Bit)

DLA

Deep Learning Accelerator

DMA

Direct Memory Access

DME

Device Management Entity

DMIC

Digital Microphone

DO

Digital Output

DP

DisplayPort

DPHY

Digital PHY

DPI

Display Parallel Interface

DPTX

DisplayPort Transmitter

DPTX AUX

DisplayPort Transmitter Auxiliary Channel

DQS

Data Strobe

DR

Dynamic Range

DRAM

Dynamic Random-Access Memory

DRAMC

Dynamic Random-Access Memory Controller

DRM

Digital Rights Management

DSD

Direct Stream Digital

DSI

Display Serial Interface

DSP

Digital Signal Processor

DSU

DynamicIQ Shared Unit

Also referred to as the "CPU cluster" or "cluster" in this document

DVFS

Dynamic Voltage and Frequency Scaling

DVI

Digital Visual Interface

E

ECAM

Enhanced Configuration Access Mechanism

ECC

Error Checking and Correction

ECRC

End to End Cycle Redundancy Check

ED

Endpoint

EDID

Extended Display Identification Data

EDMA

External Data Memory Access

EDP

Embedded DisplayPort

EEE

Energy Efficient Ethernet

EINT

External Interrupt

ELSI

Enable Line Status Interrupt

EMI

External Memory Interface

eMMC

Embedded MultiMediaCard

ENIC

Ethernet Network Interface Controller

EOL

End of List

ERBFI

Enable RX Buffer Full Interrupt

EP

Endpoint

ETBEI

Enable TX Buffer Empty Interrupt

EVB

Evaluation Board

F

FCR IIR

FIFO Control Register/Interrupt Identification Register

FD

Face Detection

FDM

Flash Data Memory

FIFO

First In First Out

FLADJ

Frame Length Adjustment Register

FM

Fast Mode

FM+

Fast Mode Plus

fps

Frames Per Second

FPU

Floating Point Unit

FSM

Finite State Machine

G

GIC

Generic Interrupt Controller

GPD

Generic Packet Descriptor

GPIO

General-purpose Input/Output

GPT

General Purpose Timer

GPU

Graphics Processor Unit

H

HCI

Host Controller Interface

HCLK

Hopping Clock

HDCP

High-bandwidth Digital Content Protection

HDMI

High-Definition Multimedia Interface

HDR

High Dynamic Range

HPD

Hot-Plug Detect

HEIF

High Efficiency Image File Format

HEVC

High Efficiency Video Coding

HLG

Hybrid Log Gamma

HPD

Hot Plug Detect

HS

High-Speed

HWO

Hardware Ownership

HSYNC/HSync

Horizontal Synchronization

HW

Hardware

I

I2C

Inter-Integrated Circuit

I2S

Inter-IC Sound

I3C

Improved Inter-IC

ICMP

Internet Control Message Protocol

IECR

Interrupt Enable Clear Register

IER

Interrupt Enable Register

IESER

Interrupt Enable Set Register

IMODI

Interrupt Moderation Interval

IOC

Interrupt On Completion

IP

Internet Protocol

IPPC

IP Power Control

IRQ

Interrupt Request

IRRX

Infrared Receiver

ISA

Instruction Set Architecture

ISP

Image Signal Processor

ISR

Interrupt Service Routine

J

JTAG

Joint Test Action Group

K

KP

Keypad

L

L1PMSS

L1 Power Management Substates

LCD

Liquid-Crystal Display

LCM

Liquid Crystal Monitor

LCR

Line Control Register

LDO

Low Dropout

LPC

Linear Pulse Code

LPF

Low-Pass Filter

LPM

Lower Power Management

LSB

Least Significant Bit

LSR

Line Status Register

LSP

Low Saturation Protection

LTR

Latency Tolerance Reporting

LTSSM

Link Training and Status State Machine

LVTS

Low Voltage Thermal Sensor

M

MACs

Multiply-Accumulate operations/Media Access Control

MBIST

Memory Built-In Self-Test

MCLK

Master Clock

MCR

Modem Control Register

MCUSYS

Microcontroller Unit System

MDC

Management Data Clock

MDIO

Management Data Input/Output

MDLA

MediaTek Deep Learning Accelerator

MIPI

Mobile Industry Processor Interface

MISO

Master Input to Slave Output

MMU

Memory Management Unit

MOSI

Master Output to Slave Input

MPP

Multi-Page Program

MPS

Maximum Packet Size

MPU

Memory Protection Unit

MSA

Main Stream Attribute

MSB

Most Significant Bit

MSDC

MMC (MultiMediaCard) and SD (Secure Digital) Controller

MSI

Message Signaled Interrupt

MSR

Modem Status Register

MTT

Multiple Transaction Translator

MUX

Multiplexer

N

NN

Neural Network

O

OTG

On-The-Go

P

PCB

Printed Circuit Board

PCIe

Peripheral Component Interconnect Express

PCLK

Peripheral Clock

PCM

Pulse Code Modulation

PD

Photo Detector

PDM

Pulse Density Modulation

PHY

Physical Layer

PHYA

Analog PHY

PHYD

Digital PHY

PIF

Processor Interface

PIO

Programmed Input/Output

PIPE

Physical Interface for PCI Express

PLL

Phase-Locked Loop

PMIC

Power Management Integrated Circuit

PMSS

Power Management Substates

POR

Power-On-Reset

PP

Page Program

PVT

Process, Voltage and Temperature

PWM

Pulse Width Modulation

PWRAP

PMIC Wrapper

Q

QE

Quad Enable

QMU

Queue Management Unit

QoS

Quality of Service

R

RBR

RX Buffer Register

RC

Root Complex

RCMM

Remote Control Multimedia

RDMA

Read Direct Memory Access

RH

Relative Humidity

RL

Read Latency

RMMI

Reference M-PHY Module Interface

RMS

Root Mean Square

ROI

Region of Interest

RPU

Region Protection Unit

RQS

Receive Queue Size

RS

Resizer

RSF

Receive Store and Forward

RSI

Reorder-able Slave Interface

RTC

Real-Time Clock

RTOS

Real-Time Operating System

RTS

Request to Send

RTSI

Request to Send Interrupt

RX

Receiver

S

S/PDIF

Sony/Philips Digital Interface Format

SA

Source Address

SAR

Successive Approximation Register

SCK

Serial Clock

SCL

Serial Clock Line

SCP

System Companion Processor/Sensor-hub Control Processer

SCR

Scratch Register

SDA

Serial Data/Serial Data Line

SDM

Sigma-Delta Modulation

SDR

Single Data Rate/Standard Dynamic Range

SDRAM

Synchronous Dynamic Random Access Memory

SFD

Start Frame Delimiter

SI

Slave Interface

SIMD

Single Instruction Multiple Data

SM

Standard Mode

SMI

Smart Multimedia Interface

SNFC

Serial NOR Flash Controller

SNFI

SPI NAND Flash Interface

SoC

System on Chip

SP

Strict Priority

SPI

Serial Peripheral Interface

SPIM

Serial Peripheral Interface Master

SPIS

Serial Peripheral Interface Slave

SPM

System Power Management

SPMC

System Power Management Controller

SPMI

System Power Management Interface

SRAM

Static Random Access Memory

SRC

Sampling Rate Converter

SS

Start Spilt

SSC

Spread Spectrum Clocking

SSUSB

SuperSpeed Universal Serial Bus

SW

Software

SYSTMR

System Timer

T

TBC

Time Base Converter

TCM

Tightly Coupled Memory

TCP

Transmission Control Protocol

TDM

Time Division Multiplexing

TE

Tearing Effect

THR

TX Holding Register

THRE

TX Holding Register Empty

TLP

Transaction Layer Packet

TMDS

Transition-Minimized Differential Signaling

TOPRGU

Top Reset Generation Unit

TOPS

Tera Operations Per Second

TQS

Transmit Queue Size

TRB

Transfer Request Block

TSMCU

Thermal Sensing Micro Circuit Unit

TSN

Time-Sensitive Networking

TT

Transaction Translator

TTC

Transmit Threshold Control

TU

Transfer Unit

TX

Transmitter

TXQEN

Transmit Queue Enable

U

UART

Universal Asynchronous Receiver/Transmitter

UDP

User Datagram Protocol

UFS

Universal Flash Storage

UI

Unit Interval

UL

Uplink

UPIU

UFS Protocol Information Unit

USB

Universal Serial Bus

UTMI

USB 2.0 Transceiver Macrocell Interface

V

VBID

Vertical Blanking ID

VBLANK

Vertical Blank

VC

Virtual Channel

VDEC

Video Decoder

VENC

Video Encoder

VESA

Video Electronics Standards Association

VLIW

Very Long Instruction Word

VP6

Tensilica Vision Processor 6

VRR

Variable Refresh Rate

VSYNC/VSync

Vertical Synchronization

W

WDT

Watchdog Timer

WEL

Write Enable Latch

WIP

Write in Progress

WPE

Warp Engine

WRR

Weighted Round Robin

X

xHC

Host Controller

xHCI

eXtensible Host Controller Interface

XIP

eXecute In Place

XOFF

Transmit Off

XON

Transmit On

ZLP

Zero Length Packet

3 Features Description

The MT8395 device is a highly-integrated, powerful platform designed for a wide range of Artificial Intelligence (AI) and Internet of Things (IoT) use cases requiring high-performance edge processing, advanced multimedia and connectivity capabilities, multiple high-resolution cameras, connected touchscreen displays, and use of a multi-tasking High-Level Operating System (HLOS).

The highly-capable octa-core application processor utilizes the Arm® DynamIQ™ technology by combining high-performance Cortex-A78 and power-efficient Cortex-A55 cores, equipped with Arm Neon™ engine. The application processor offers the necessary processing power to support the latest OpenOS, along with its demanding applications such as smart home appliance, industrial IoT and other AI embedded devices. This content can be enhanced by the 2D/3D graphics accelerator (Arm Mali-G57 MC5 GPU) and then visualized on a high-resolution touchscreen display. To provide advanced multimedia applications and services such as streaming audio and video, the device features multi-standard video encoder and decoder engines, and an advanced audio subsystem.

The dual-core AI Processor Unit (APU) enables deep learning, Neural Network (NN) acceleration, and Computer Vision (CV) applications. The latter, combined with the up to 48MP camera, can clearly and accurately perform AI-vision functions such as facial recognition, object identification, scene analysis, optical character recognition and much more.

An extensive set of interfaces, connectivity, flexible storage and memory options further enhance the capabilities of the device and give product designers freedom to customize.

3.1 Application Processor

3.1.1 Overview

The **MCUSYS** (the application processor subsystem) is responsible for running the operating system and application programs, providing different levels of power efficiency and computing power to satisfy a wide range of system power and performance requirements. It is composed of:

- A local bus fabric (CPUBIU)
- An interrupt controller (GIC-600)
- A CPU cluster (DSU with the CA78 and CA55 cores)

The Cortex-A55 cores are specifically optimized for power efficiency to minimize the power consumption for daily usage scenarios and lightweight applications, while the Cortex-A78 cores are designed for performance-driven applications, and providing the best user experiences for heavy tasks.

The MCUSYS supports DVFS technology to allow the CPU to run at different frequencies and voltage configurations depending on the application requirements. Additionally, the power of each CPU core can be individually turned off when not in use. In the standby mode, the MCUSYS can be completely shut down to further reduce power consumption.

3.1.2 Features

- 4 × Arm Cortex-A55 cores with 32KB I/D cache and 128KB L2 cache
- 4 × Arm Cortex-A78 cores with 64KB I/D cache and 256KB L2 cache
- 2MB L3 cache
- Interrupt controller, Arm GIC-600
- Advanced DVFS mechanism

3.1.3 Block Diagram

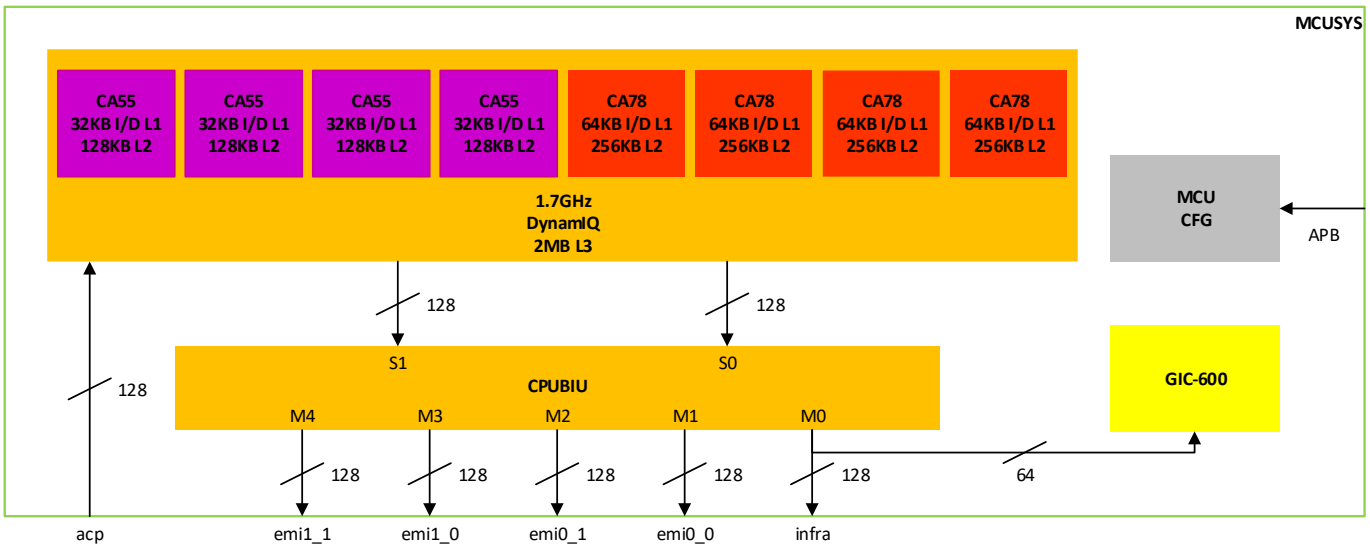


Figure 3-1 Block Diagram of MCUSYS

- The MCUSYS consists of
  - 4 Arm Cortex-A55 cores
  - 4 Arm Cortex-A78 CPU cores
- The cores are integrated into the DynamicIQ Shared Unit (DSU), also known as the “CPU cluster” or “cluster”.
- The DSU has a snoop control unit to manage data coherency, while a 2MB L3 cache is shared with the CPU cores.
- The GIC-600 manages interrupts between the CPU cores and systems.
- The Accelerate Coherency Port (ACP) interface provides coherent transactions support between the MCUSYS and other execution engines such as the GPU. The debug APB interface connects to the MCUSYS debug logic providing the debug functions.
- The MCUSYS utilizes the AXI to access system memory and devices.

3.1.4 Function Description

The CPU cluster service application execution and operation systems collaborate with the advanced DVFS mechanism to archive a balance between power and performance for a wide range of workloads.

The CPUBIU routes transactions to dedicated memory channels, reducing path latency and improving access performance, but it does not support coherency functions. The GIC-600 supports and manages interrupts in the MCUSYS for interrupt masking, prioritization, and security. Additionally, the debug logic of the MCUSYS supports static debugging to connect to the system via certain interfaces (e.g. via the JTAG), allowing cross-triggering of internal and external trigger events for debugging.

3.1.5 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.1.6 References

- Arm® Cortex®-A55 Core Technical Reference Manual
- Arm® Cortex®-A78 Core Technical Reference Manual
- Arm® DynamIQ™ Shared Unit Revision: r4p1 Technical Reference Manual
- Arm® CoreLink GIC-600 Generic Interrupt Controller Technical Reference Manual

3.2 Graphics Accelerator

3.2.1 Overview

The device graphics accelerator (GPU) is based on Arm Mali-G57 MC5 core. It is used to process extremely complicated graphics and perform general processing tasks assigned by the main application processor.

3.2.2 Features

The GPU supports the following key features:

- An enhanced API feature set with high-performance support for both shader-based and fixed-function graphics APIs. The supported graphics and compute API industry standards are:
  - OpenGL ES 1.1, 2.0, 3.2
  - Vulkan 1.0, 1.1
  - OpenCL 1.0, 1.1, 1.2, 2.0, 2.1, 2.2
- Anti-aliasing capabilities
- An effective core for General Purpose computing on Graphics Processing Units (GPGPU) applications
- High memory bandwidth and low-power consumption for 3D graphics content
- Arm Frame Buffer Compression (AFBC), version 1.3
- Adaptive Scalable Texture Compression (ASTC)
- 8-bit, 10-bit and 16-bit YUV input and output formats
- Secure processing of Digital Rights Management (DRM) protected content
- Bus protocol:
  - 4 × 128-bit Arm AMBA® 4 ACE master interface for external memory access
  - 1 × 32-bit Arm AMBA 4 AXI slave interface for GPU configuration
- L2 cache:
  - 4 banks × 256KB
  - 4-way, set-associative
- Cache coherency support:
  - Within GPU
  - Between GPU and other system resources

3.3 Digital Signal Processor

3.3.1 Overview

The Digital Signal Processor (DSP) is responsible for running the operating system and application programs. It comprises:

- Single-core Cadence HiFi 4 Audio Engine DSP
- AXI3 bus interface unit (in-house MediaTek bus)
- System Power Management Controller (SPMC)

The Cadence HiFi 4 DSP is a highly optimized audio processor geared for efficient execution of audio and voice codecs and pre- and post-processing modules.

The processor goes beyond the HiFi 3 with support for four 32x32-bit MACs, some support for 72-bit accumulators, limited ability to support eight 32x16-bit MACs, a fourth VLIW (Very Long Instruction Word) slot and the ability to issue two 64-bit loads per cycle. There is an optional floating point unit available, providing up to four single-precision IEEE floating point MACs per cycle. The extra resources provide significant performance improvements compared with the HiFi 3, particularly on pre/post-processing algorithms.

The HiFi 4, a Single Instruction Multiple Data (SIMD) processor, has the ability to work in parallel on two 32-bit data items or four 16-bit data items. HiFi 4 is a VLIW architecture, supporting the execution of four operations in parallel.

3.3.2 Features

- **Single-core Cadence HiFi 4 DSP operates at 720 MHz (0P75V),** including:
  - 32KB L1 I-cache
  - 128KB L1 D-cache
  - Data retention not supported by Pre-fetch buffer, I-cache, D-cache, ITag, or DTag Joint Test Action Group (JTAG)
  - 18 interrupts configurable by intc
  - Supports the SPM to control power sequence.

- **Peripheral:**
  - 256KB L2TCM
  - One UART
  - Five mailboxes
  - Eight semaphores
  - One system timer
  - One watchdog timer

3.3.3 Block Diagram

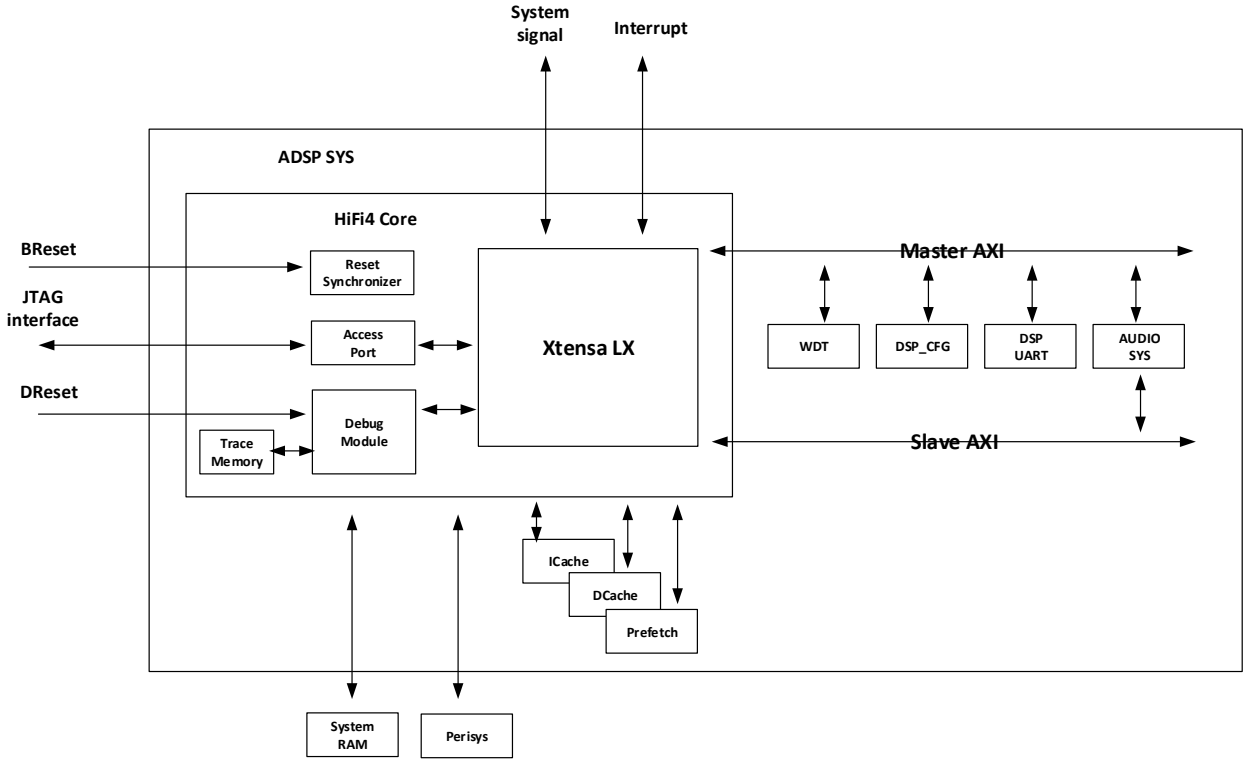


Figure 3-2 HiFi 4 DSP Block Diagram

3.3.4 Function Description

3.3.4.1 Pipeline Stages

The DSP consists of a 7-stage pipeline with two-cycle latency for both the instruction-memory fetch and the data-memory fetch.

- Pipeline length 7
- Instruction memory fetch latency 2
- Cycle of execute stage 1
- Cycle of modify stage 3
- Cycle of write-back stage 4

3.3.4.2 Layer 1 Cache Controller

The host processor of the DSP includes a cache controller that has a 32KB instruction cache and a 128KB data cache implemented to improve the code/data fetch performance.

To further enhance cache performance, the data cache supports write-back operations. Additionally, the data cache can be switched programmatically between write-back and write-through modes.

Both the instruction cache and data cache support dynamic-cache-way-disable. This feature allows for the independent disabling and re-enabling of cache ways in both caches. It also facilitates power saving, as you can clean cache ways before

disabling them and initialize cache ways when enabling them. When a cache way is disabled, the cache memory block from service is removed, and therefore the total cache capacity is reduced by “1/(number of ways in service)”.

- Instruction cache details
  - Instruction cache size in bytes 32768
  - Instruction cache ways 4
  - Instruction cache line size bytes 128
  - Dynamic way disable Supported

- Data cache details
  - Data cache size in bytes 131072
  - Data cache ways 4
  - Data cache line size bytes 128
  - Data cache write-back Supported
  - Dynamic way disable Supported

3.3.4.3 Memory Management

The Memory Protection Unit (MPU) provides memory protection functionality with a smaller footprint than the Linux-compatible Memory Management Unit (MMU) while offering greater flexibility and features than the Region Protection Unit (RPU).

The features of MPU are as follows:

- A configurable number (16 or 32) of entries that determine the regions
- A configurable address granularity (minimum: 4KB)
- 12 choices of access-rights settings per region
- More than 200 different memory-type choices per region, including the ability to specify the internal L1 cache behavior, which is different from an external (AXI or PIF) cache behavior (attributes)
- Identity map implementation: No address translation
- Runtime modifiable foreground memory map
- Static background memory map
- Unified instruction and data memory maps

3.3.4.4 Interrupt

The DSP supports 32 maskable interrupts with four priority levels.

Table 3-1 Intc Interrupt

Interrupt	Type	Level	BInterrupt Pin	Note
0	Level	1	0	I irq_ext[730:0]
1	Level	1	1	cpu2dsp_irq | audio2dsp_irq | uart2dsp_irq | cq_gdma_irq[2:0]
2	Level	1	2	dsp_timer_irq[0]
3	Level	1	3	dsp_timer_irq[1]

Interrupt	Type	Level	BInterrupt Pin	Note
4	Level	1	4	dsp_timer_irq[2]
5	Level	1	5	dsp_timer_irq[3]
6	Edge	1	6	N/A
7	Edge	1	7	N/A
8	Level	2	8	I irq_ext[730:0]
9	Level	2	9	cpu2dsp_irq | audio2dsp_irq | uart2dsp_irq | cq_gdma_irq[2:0]
10	Level	2	10	dsp_timer_irq[0]
11	Level	2	11	dsp_timer_irq[1]
12	Level	2	12	dsp_timer_irq[2]
13	Level	2	13	dsp_timer_irq[3]
14	Edge	2	14	N/A
15	Edge	2	15	N/A
16	Timer.0	1	-	Timer 0
17	Timer.1	2	-	Timer 1
18	Timer.2	3	-	Timer 2
19	WriteErr	1	-	WriteErr
20	Profiling	1	-	Profiling
21	SW	1	-	Software
22	SW	2	-	Software
23	Level	1	16	N/A
24	Level	2	17	N/A

3.3.5 Theory of Operations

- HiFi 4 DSP User Guide
- Xtensa LX7 Microprocessor Data Book

3.3.5.1 Debug Monitor Register

The DSP contains a set of debug monitor registers that enable the monitoring of the internal signals within the DSP core for debugging purposes. For more information, please refer to Xtensa Debug Guide.

Table 3-2 Debug Signals

Signal	I/O	Description	Dsp0cfg_reg
PDebugEnable	Input	Trace enable	PDebugBus0[0]
PDebug_latch_en	-	-	PDebugBus0[4]
PDebugData[31:0]	Output	Trace debug data	PDebugData
PDebugInbPIF[7:0]	Output	Inbound PIF transaction information	PDebugBus1[31:24]
PDebugOutPIF[7:0]	Output	Outbound PIF transaction information	PDebugBus1[23:16]
PDebugInst[31:0]	Output	Instruction information	PDebugInst

Signal	I/O	Description	Dsp0cfg_reg
PDebugLS0Stat	Output	LSU0 status	PDebugLS0Stat
PDebugLS1Stat	Output	LSU1 status	PDebugLS1Stat
PDebugPC	Output	Trace program counter	PDebugPC
PDebugPrefetchL1Fill[3:0]	Output	Prefetch L1 fill information	PDebugBus0[19:16]
PDebugPrefetchLookup[7:0]	Output	Prefetch lookup information	PDebugBus1[15:8]
PDebugStatus[7:0]	Output	Trace status information	PDebugBus1[7:0]

3.3.6 Power Management

3.3.6.1 Constraint to Access DSP Configuration Register

The DSP configuration register is in the Always-On power domain.

3.3.6.2 Sleep and Wakeup

1. After the *WAITI* instruction is completed, the DSP sets the interrupt level in the *PS.INTLEVEL* register to the value encoded by the instruction.
2. The processor then waits for all processor memory interfaces to become idle and asserts the *DSP_PWaitMode* signal.
3. During this time, all processor operations are suspended until a non-masked interrupt occurs.

Note:

- Please refer to Xtensa LX7 Microprocessor Data Book 22.5.

3.3.7 Programming Guide

3.3.7.1 Basic Setting

- HiFi 4 DSP User Guide
- Xtensa LX7 Microprocessor Data Book

3.3.7.2 Basic Setting Flow

Basic Setting

1. Load instruction code and data code to 0x1084_0000~0x1087_FFFF (256KB L2TCM)
2. Change the DSP clock to an at-speed clock, with the default speed of 26 MHz.
   a. CLK_CFG_8 = 0x02000000; *CLK_CFG_UPDATE1 = 0x8; change clock from DSPPLL. (DSPPLL_CON1 can modify DSPPLL frequency)
3. PWR ON DSP PWR
4. Set AltResetVec to the default value 0x40000000 (If needed)
5. Set Reset_sw = 0x1b
   a. Set StatVectorSel (Reset_sw[4]) to 1
   b. Keep RunStall (Reset_sw[3]) to 1 (default)

c. Keep PReset_sw (Reset_sw[2]) to 0
d. Set DReset_sw (Reset_sw[1]) to 1
e. Set BReset_sw (Reset_sw[0]) to 1

6. Set Reset_sw = 0x18
   a. Keep StatVectorSel (Reset_sw[4]) to 1
   b. Keep RunStall (Reset_sw[3]) to 1 (default)
   c. Keep PReset_sw (Reset_sw[2]) to 0
   d. Set DReset_sw (Reset_sw[1]) to 0
   e. Set BReset_sw (Reset_sw[0]) to 0
7. Set Reset_sw = 0x10
   a. Keep Set StatVectorSel (Reset_sw[4]) to 1
   b. Set RunStall (Reset_sw[3]) to 1 (release run_stall)
   c. Keep PReset_sw (Reset_sw[2]) to 0
   d. Keep DReset_sw (Reset_sw[1]) to 0
   e. Keep BReset_sw (Reset_sw[0]) to 0

3.3.8 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.3.9 Reference

- HiFi 4 DSP User Guide
- Xtensa LX7 Microprocessor Data Book
- Xtensa Debug Guide

3.4 AI Processor Unit

3.4.1 Overview

The MediaTek AI Processor Unit System (APUSYS) significantly enhances multimedia performance by exhibiting remarkable computing capabilities. The key components of the APUSYS are:

- One 2-core programmable Tensilica **Vision Processor 6 (VP6)**, for both the traditional Computer Vision (CV) algorithms and NN algorithms.
- One 2-core **MediaTek Deep Learning Accelerators (MDLA)**, for the Neural Network (NN) algorithms.

The hardware design is specifically optimized for job allocation between the NN (MDLA) and CV (VP6) engines, resulting in efficient performance. The External Direct Memory Access (EDMA) engine supports compression/decompression, data movement and format conversion.

The internal APU Tightly Coupled Memory (TCM) reduces external DRAM bandwidth and minimizes data transmission latency. For external interface parts, there are four AXI buses to access the external DRAM. The APUSYS exchanges data with other subsystems through the external DRAM.

3.4.2 Features

- **The VP6 supports both the AI and CV.**
  - Dual core to support simultaneously multiple applications
  - Per core configuration
    - L1 Instruction memory per core: 64 KB + 128 KB cache
    - L1 data memory per core: 128 KB + 128 KB
    - vFPU to support high-precision requirement applications
  - Top performance:
    - Fix 8: 0.85 TOPs,
    - Fix 16: 0.21 TOPs,
    - FP16: 0.11 TOPs,
    - FP32: 0.05 TOPs

- **The MDLA supports high-computation demands NN applications.**
  - Peak power reduction
  - Top performance:
    - 8(A) x 8(W) : 3.9 TOPS
    - 16(A) x 16(W): 2.0 TOPS
    - FP16 : 2.0 TOPS
  - Simultaneous pipelined hardware function block (CONV/ACT/POOL/EWE/BILINEAR)
  - Enhancement of layer fusion to further reduce DRAM/TCM memory bandwidth
  - Supports Android NN asymmetric quantized data format.
  - Supports compressed weight to reduce DRAM bandwidth.
  - Supports Android NN asymmetric quantized data format.
  - Supports compressed weight to reduce DRAM bandwidth.

- **The EDMA supports data movement and data format conversion.**
  - Format conversion and data movement between memories

- **Memory subsystem in APUSYS**
  - 1 MB TCM as the 2nd level data memory

3.4.3 Block Diagram

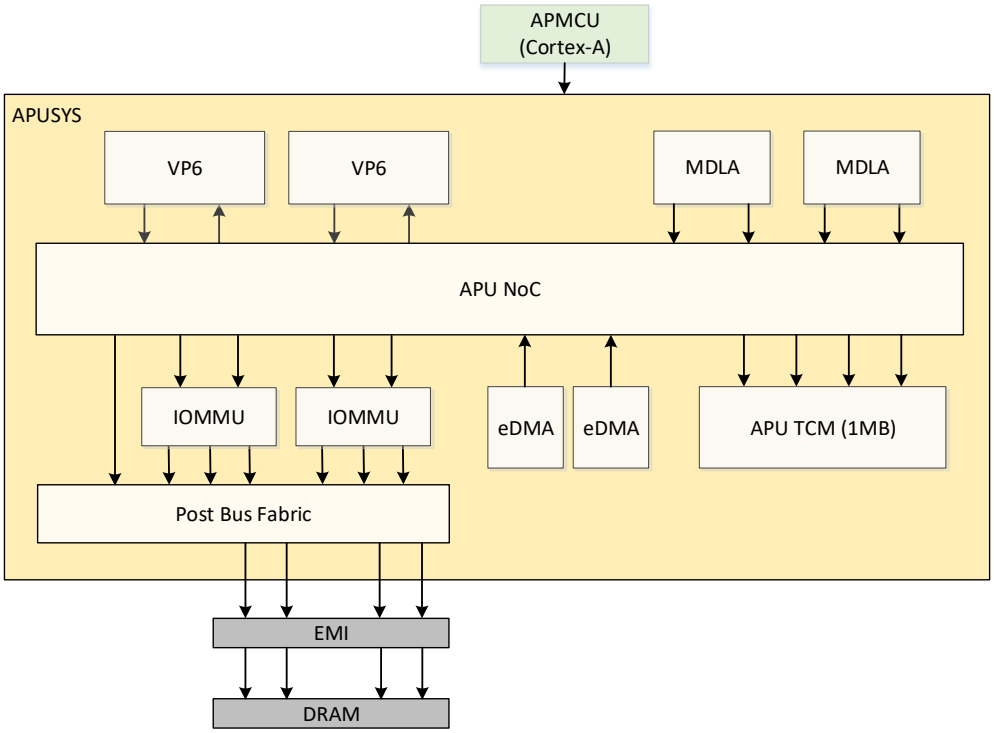


Figure 3-3 Block Diagram and Interface of APUSYS

3.4.4 Function Description

3.4.4.1 VP6

3.4.4.1.1 Overview

The Tensilica Vision-P6 DSP (VP6) is a programmable AI processor unit with high-performance and high-flexibility for vision applications. There are two VP6 in APUSYS.

Each VP6 has a 5-slot Very Long Instruction Word (VLIW) architecture. To achieve high-performance computing, it utilizes the multiply-accumulate (MAC) unit to compute up to 256 8 bits x 8 bits MACs in one cycle. To achieve high-throughput bandwidth for computing, it uses up to “2-slot 64-byte loads” or “1-slot 64-byte load and 1-slot 64-byte store” to access 128-byte data bandwidth in one cycle.

Each VP6 is a vector Single Instruction Multiple Data (SIMD) DSP. For high-flexibility programming, it supports 64-way 8-bit operations, 32-way 16-bit operations, and 16-way 32-bit operations in fixed-point format. It also supports 16-way single-precision operations and 32-way half-precision operations in floating-point format.

Each VP6 supports special features. The Scatter/Gather operations enhance random accessing to local Data RAM, and the histogram operations accelerate binning function.

3.4.4.1.2 Features

Table 3-3 shows the features of a single VP6.

Table 3-3 Features of a Single VP6

Feature	Configuration
Instruction RAM	64KBytes
Instruction Cache	128KBytes
Data RAM	256KBytes
Number of MACs in fixed-point operations	8 bits x 8 bits: 256 (64-way multiply or multiply-accumulate quad operations) 8 bits x 16bits: 128 (64-way multiply or multiply-accumulate pair operations) 16bits x 16bits: 64 (32-way multiply or multiply-accumulate pair operations) 32 bits x 16 bits: 16 (16-way multiply or multiply-accumulate operations)
Number of MACs in floating-point operations	16 bits x 16 bits: 32 (32-way multiply or multiply-accumulate operations) 32 bits x 32 bits: 16 (16-way multiply or multiply-accumulate operations)
Half-precision operations	32-way half precision vector FPU
Single-precision operations	16-way single precision vector FPU
Histogram operations	Native histogram instructions
Scatter/Gather operations	32-way 16-bit vector elements accessing to local Data RAM
iDMA	64 outstanding re-order buffers
User TIE	MTK_TIE2APB: Accessing ability to APB bus

Note:

- TIE denotes “Tensilica Instruction Extension”.
- The User TIE “MTK_TIE2APB” is used to support APB master in Vision-P6 DSP. The programmer can use API functions such as “IPU_SendAPBWrite(data,addr)” and “IPU_SendAPBRead(data,addr)” to use APB transaction to access configuration registers.

3.4.4.1.3 Block Diagram

Figure 3-4 shows the block diagram of a single VP6.

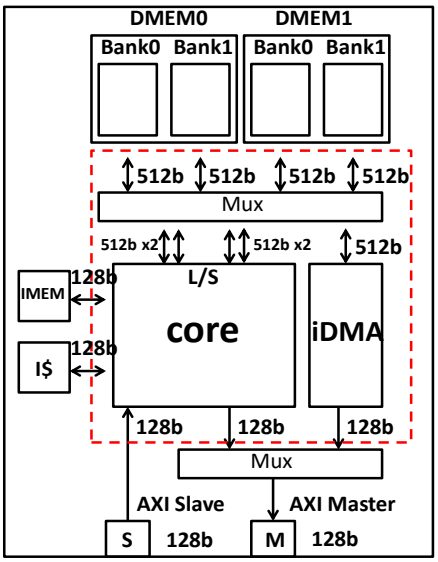


Figure 3-4 VP6 Block Diagram

3.4.4.2 MDLA

3.4.4.2.1 Overview

The MediaTek Deep Learning Accelerator (MDLA) comprises three primary components and handle the following tasks:

- **Command engine (CMDE)**: Decodes commands and maps them to the corresponding registers. It receives input commands and converts them into a format that is interpretable by other components of the MDLA.
- **Convolutional engines** and **non-convolutional engines**: Operate on the input to efficiently generate the desired tensor data.

3.4.4.2.2 Features

- **Convolutional engine**
  - Convolutional 2D
  - Depth-wise convolutional
  - Fully connected
  - Transpose convolutional
  - Dilated convolutional
- **Element-wise engine**
  - Element-wise functions:
    - BN, Mul, Add, Sub, Max, Min, Abs, Neg, Sqr, IN_sub
  - Activation functions:
    - ReLu, ReLu1, ReLu6, PreLu, Sigmoid, Tanh, Elu, GeLu, Exp, Rcp, Sqrt, Mish, Rsqrt
- **Pool engine**
  - Pooling functions:
    - Global pooling, Local pooling
  - Resize function
    - Resize bilinear, resize nearest (nearest-neighbor interpolation), resize nearest floor
- **Transpose engine**
  - C-W transpose
  - Depth2Space
  - Space2Depth
- **Store engine**: Reshape

3.4.4.2.3 Block Diagram

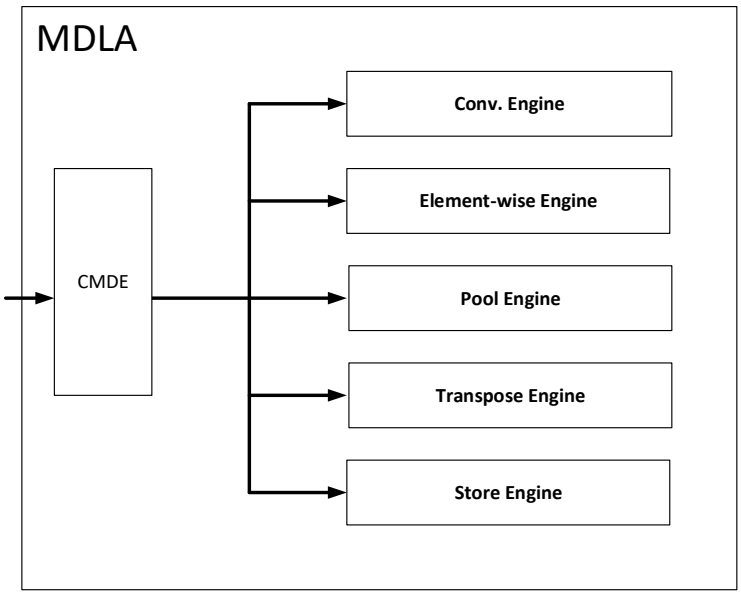


Figure 3-5 Key Block Diagram of MDLA

3.4.4.3 EDMA

3.4.4.3.1 Overview

The External Direct Memory Access (EDMA) serves the exchange and conversion of data formats between the APUSYS and other subsystems or engines. This allows the computing engines, such as the MDLA, VP6, to avoid the negative effects of inconsistent data formats from other subsystems. Consequently, the exchanged data format within the APUSYS is more uniform and efficient.

3.4.4.3.2 Features

- **Normal functions:**
  - Data copy
  - Fill constants
- **Numeric conversion**
  - F16 to F32
  - F32 to F16
- **Format conversion**
  - RGGB to Bayer
  - Bayer to RGGB
- **Image compression/decompression**
  - AFBC_RGBA_U8 to RGBA_8888
  - AFBC_YUV420_U8 to YUV4444_U8
  - RGBA_8888 to AFBC_RGBA_U8
  - YUV420_U8 to AFBC_YUV420_U8
  - YUV444_U8 to AFBC_YUV420_U8

3.4.5 Theory of Operations

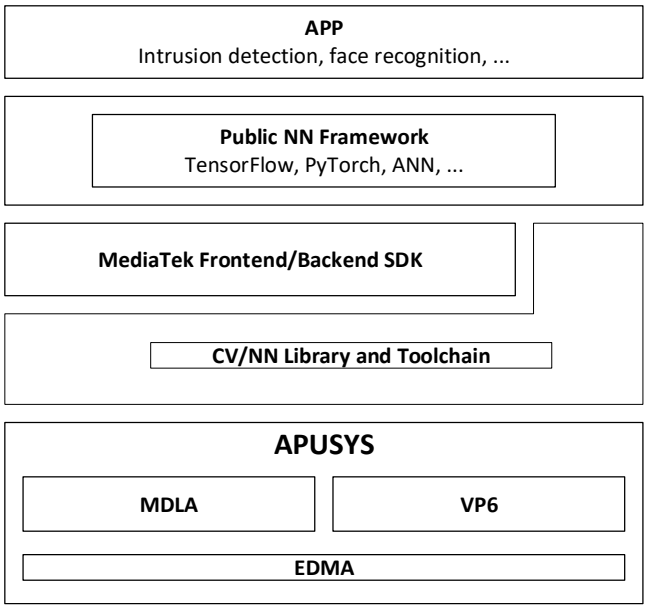


Figure 3-6 APU Theory of Operation Construction

To enable effective uses of the APU, public frameworks provide a reliable foundation, on which software can be built. These frameworks offer a structured approach to application development and are designed to simplify the process. Several open-source NN frameworks, including TensorFlow, PyTorch, and ANN, are used to construct Software Development Kits (SDK), which provide APIs and libraries that allow developers to access and control both the MDLA and VPU hardware.

3.4.6 Programming Guide

NeuroPilot is a renowned software framework developed by MediaTek. It incorporates various public frameworks for application development, including the VPU and MDLA software stacks. Additionally, the EDMA is utilized for efficient data transmission. While the VPU and MDLA begin their functionality through the thread dispatcher to instruction buffer or CMDE, programming is solely possible through the MediaTek SDK. For more detailed information, please refer to the NeuroPilot website.

3.5 System Companion Processor

3.5.1 Overview

The System Companion Processor (SCP) is a processor subsystem that includes two MDSP RV33 processors and a variety of peripherals. The special design of the SCP makes it suitable for running applications such as voice wake-up, sensor hub and future tasks when the entire device is in suspend mode. The SCP is connected to an infra bus and therefore can access DRAM, audio SRAM and other hardware resources through it.

3.5.2 Features

3.5.2.1 MDSP RV33 Features

The MDSP RV33 is a low-power DSP for sensor, voice and audio applications. It supports the following key features:

- 32-bit integer core
- Single-precision Floating Point Unit (FPU)
- RISC-V compatible instruction set
- 6-stage pipeline
- L1 cache memory system:
  - 32KB I-cache
  - 32KB D-cache
- AXI and prefetch

3.5.2.2 SCP Features

The SCP supports the following key features:

- 2 x MDSP RV33 cores
- 32KB Shared L1 TCM
- 768KB Shared L2 TCM
- An AXI master interface for device memory and register access
- An AXI master interface for device DRAM access
- An AHB master interface for audio memory and register access
- An AXI slave interface for configuration by MCUSYS
- Interprocessor interrupt to MCUSYS
- Serial audio interface to support Voice wake-up
- 6 × 32-bit down-count timers per core, with selectable clock source
- Interrupt controller per core
- Support of 12-input external interrupt with debounce function
- 2 × I2C
- 2 × I3C
- 3 × SPI
- 2 × UART
- 8-bit GPIO
- Direct path to PMIC wrapper
- Watchdog timer

3.5.3 Block Diagram

Figure 3-7 shows the SCP system and the related connection with AP SoC.

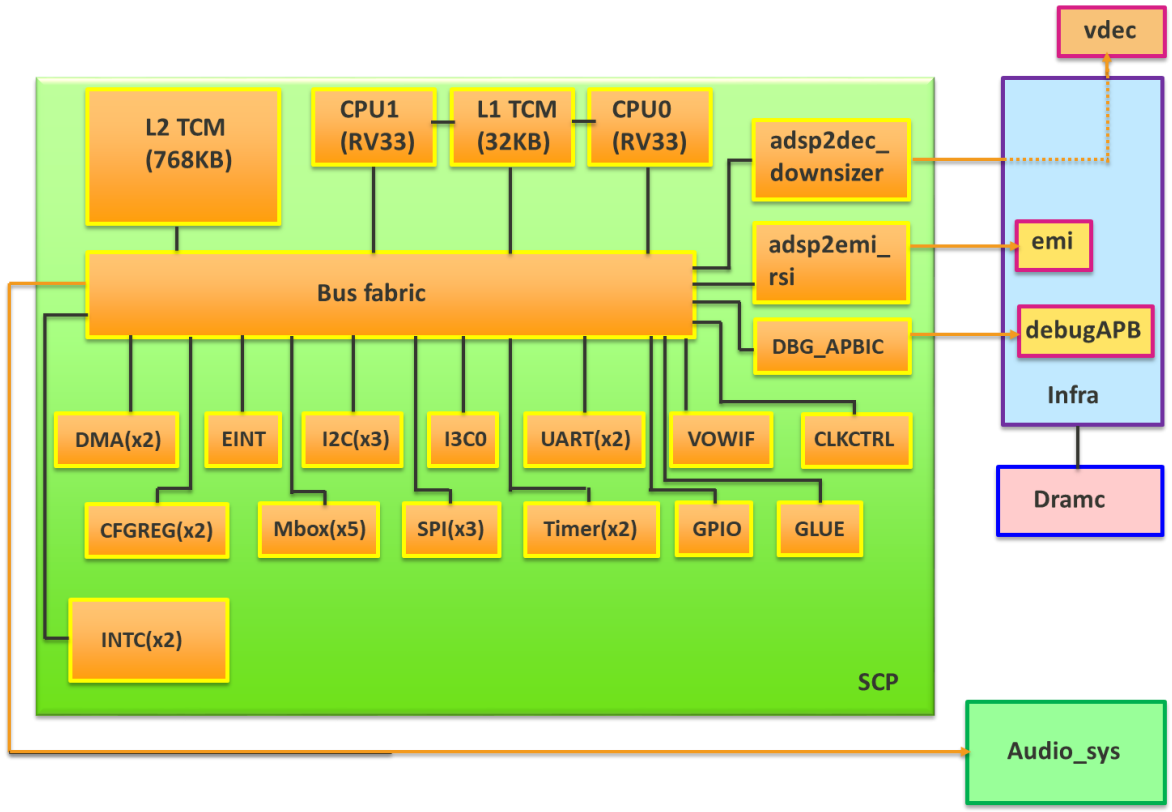


Figure 3-7 Block Diagram of SCP

Table 3-4 Frequency

DVFS Level	SCP Frequency (MHz)	SCP Clock Source
Turbo	416	26 MHz + PLL
High performance	280	ULPOSC
Low power	248	ULPOSC
Ultra-low power	182	ULPOSC

3.5.4 Function Description

3.5.4.1 Overview of MD32-RV

The MRV33E252 is a low-power DSP for sensor, voice and audio applications. The ISA (Instruction Set Architecture) is based on free and open ISA “RISC-V” and customized MediaTek proprietary DSP instructions for differentiation.

MD32RV is designed with the necessary peripherals for embedded applications. It consists of processor core, level 1 memory subsystem and memory protection unit. The features of MD32-RV processor are as follows:

- 32-bit integer core
- Single precision FPU
- 1/D L1 cache and TCM
- Memory protection unit
- AXI bus interface
- Vectored interrupt controller

- 32-bit down count timer
- Performance monitor
- JTAG embedded ICE
- APB debug interface
- Debug cross-trigger interface

MD32RV core is a 32-bit integer core and single-precision FPU:

- RISC-V compatible instruction set
- RV32 Int/Mul/Atomic/Floating
- Privilege mode
- Standard control and status registers
- Standard interrupt and exception
- Little endian
- The external interface for MD32RV processor contains:
- Instruction AXI bus and Data AXI bus (AMBA 3 AXI protocol)
- The APB interface could be used to debug the software through an on-chip bus.
- Support conventional JTAG ice for run control and debug, which depends on configuration
- 32-256 interrupt sources for external devices to trigger interrupt requests
- 8 fast interrupt sources for external devices to trigger interrupt requests.

Interface:

- System Clock and Reset: CLK/RST
- Interrupt: External, Fast, SysTimer Interrupts
- Debug interface: JTAG, Debug, APB, Cross Trigger
- Performance and Debug Monitor: Output PMU, Core status for monitor, CG/Halt status
- Pre-defined Region: Low power access TCM/CACHE
- AXI
- SRAM: ITCM/DTCM/ICACHE/DCACHE/TBUF

Micro architecture:

- 6-stage pipeline, IF/PD/ID/E1/E2/WB
- 3 execution pipelines (2 integer engines and 1 FPU engine)
- Dynamic branch predication
- Dual MACs
- Single load/store
- Non-blocking hardware divider
- Halt for interrupt instruction

3.5.4.2 RSI

The RSI is the abbreviation of Reorder-able Slave Interface. The so-called Slave Interface (SI) is an AXI dispatcher circuit that has one AXI slave interface and N master interfaces. The SI re-routes the incoming traffic from the slave interface to one of the N master interfaces according to its address.

The RSI is more specifically a 1-to-2 dispatcher according to a programmable dispatching rule. Its major purpose is to dispatch an aggregated AXI bus to two different DRAMs by each subsystem.

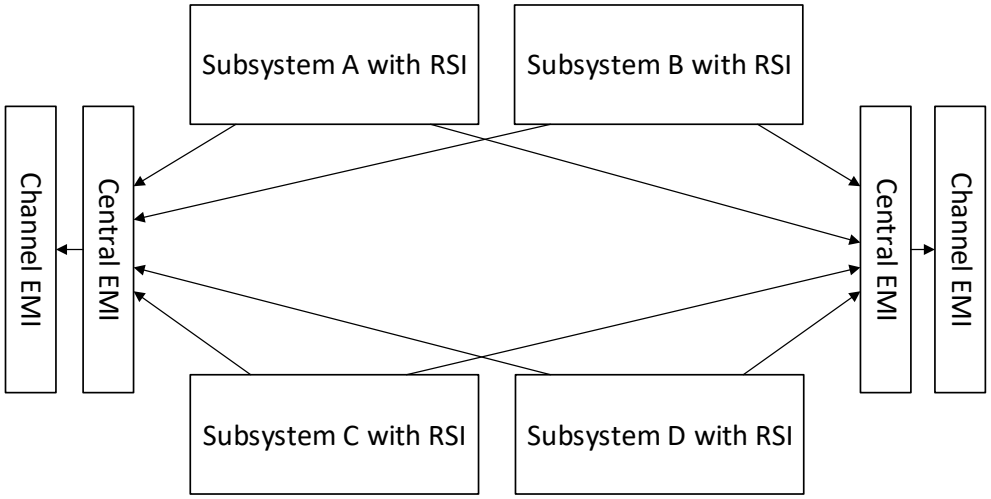


Figure 3-8 Architecture of RSI

3.5.4.2.1 RSI Features

- Two dispatch modes:
  - One mode is to dispatch based on the result of selectively XOR of address bit[11:8] by EN[3:0]
    - If ^(EN[3:0] & address[11:8]) = 0, dispatch to master interface #0.
    - If ^(EN[3:0] & address[11:8]) = 1, dispatch to master interface #1.
  - The other mode is always to dispatch transaction to master interface #1.
- Configurable outstanding number of commands.
- Same AXI-ID can have outstanding commands to two master interfaces.
- Automatically split an incoming command into two commands when this command crosses master interfaces.

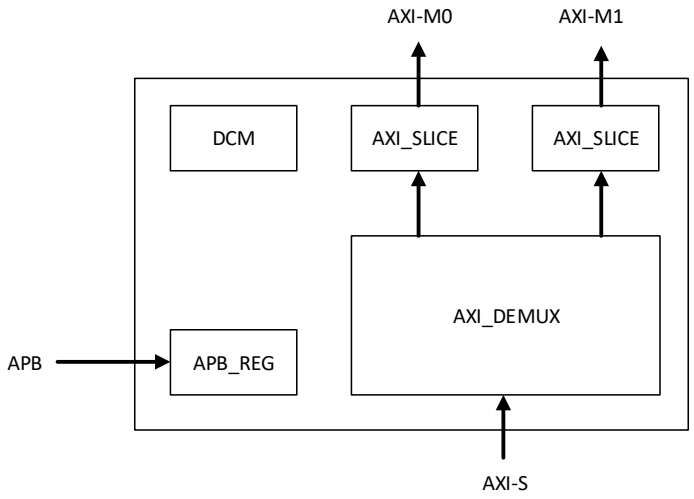


Figure 3-9 RSI Block Diagram

The AXI_DEMUX implements the major function of the RSI. It dispatches commands according to the address mapping rule configured in RSI. The mapping rule is to dispatch by selective XOR of address bit [8], [9], [10] and [11]. Each of the four bits is programmable to be enabled to participate in the XOR function.

DCMs are implemented in the RSI to reduce the unnecessary clock toggles for power-saving purposes. There are two DCMs to gate read and write clocks respectively when RSI is idle or stalled.

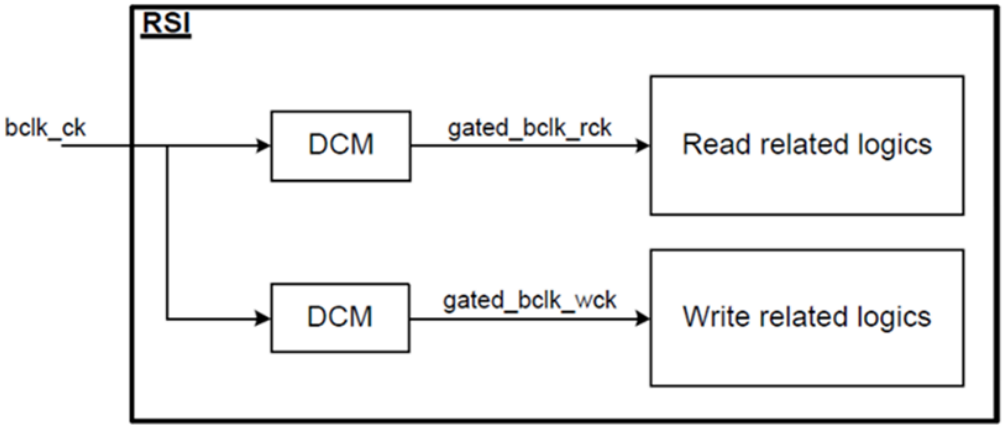


Figure 3-10 RSI DCM Block Diagram

Dispatching to Master Interface #1:

It is easy to dispatch all the transactions to master interface #0 by setting the interleaving rule to be 4'b0. However, there is no method to dispatch all the transactions to master interface #1 by setting the interleaving rule. To make the design more flexible, the RSI supports dispatching all transactions to master interface #1 by using the input port “ext_force_m1”.

3.5.5 SCP Signal Descriptions

Table 3-5 presents SCP signal descriptions.

Table 3-5 SCP Signal Descriptions

Signal Name	Type	Description	Ball Location
SCP_I3C0(1)			
SCP_SCL0	DIO	SCP I3C clock 0	AM28
SCP_SDA0	DIO	SCP I3C data 0	AK27
SCP_I2C1(1)			
SCP_SCL1	DIO	SCP I2C clock 1	AL28, AL27
SCP_SDA1	DIO	SCP I2C data 1	AM29, AL29
SCP_I2C2(1)			
SCP_SCL2	DIO	SCP I2C clock 2	AU18, AL9
SCP_SDA2	DIO	SCP I2C data 2	AT18, AM10
SCP_I3C3(1)			
SCP_SCL3	DIO	SCP I3C clock 3	AT11
SCP_SDA3	DIO	SCP I3C data 3	AL10

Signal Name	Type	Description	Ball Location
SCP_SPI0			
SCP_SPI0_CK	DO	SCP SPI0 serial clock	F32
SCP_SPI0_CS	DO	SCP SPI0 chip select	F31
SCP_SPI0_MI	DI	SCP SPI0 master input/slave output	G31
SCP_SPI0_MO	DO	SCP SPI0 master output/slave input	G32
SCP_SPI1_A			
SCP_SPI1_A_CK	DO	SCP SPI1 serial clock	AU23
SCP_SPI1_A_CS	DO	SCP SPI1 chip select	AT22
SCP_SPI1_A_MI	DI	SCP SPI1 master input/slave output	AR23
SCP_SPI1_A_MO	DO	SCP SPI1 master output/slave input	AT23
SCP_SPI1_B			
SCP_SPI1_B_CK	DO	SCP SPI1 serial clock	AP14
SCP_SPI1_B_CS	DO	SCP SPI1 chip select	AL12
SCP_SPI1_B_MI	DI	SCP SPI1 master input/slave output	AR12
SCP_SPI1_B_MO	DO	SCP SPI1 master output/slave input	AK12
SCP_SPI2			
SCP_SPI2_CK	DO	SCP SPI2 serial clock	AP17
SCP_SPI2_CS	DO	SCP SPI2 chip select	AT18
SCP_SPI2_MI	DI	SCP SPI2 master input/slave output	AL17
SCP_SPI2_MO	DO	SCP SPI2 master output/slave input	AU18
SCP_UART			
TP_UCTS1_AO	DI	SCP UART1 clear to send (active low)	AT10
TP_UCTS2_AO	DI	SCP UART2 clear to send (active low)	AP24
TP_URTS1_AO	DO	SCP UART1 request to send (active low)	AM8
TP_URTS2_AO	DO	SCP UART2 request to send (active low)	AM25
TP_URXD1_AO	DI	SCP UART1 receive data	AN8
TP_URXD2_AO	DI	SCP UART2 receive data	AN24
TP_UTXD1_AO	DO	SCP UART1 transmit data	AR8
TP_UTXD2_AO	DO	SCP UART2 transmit data	AN25
SCP_GPIO			
TP_GPIO0_AO	DIO	SCP GPIO0	H31
TP_GPIO1_AO	DIO	SCP GPIO1	G34
TP_GPIO2_AO	DIO	SCP GPIO2	J31
TP_GPIO3_AO	DIO	SCP GPIO3	G33
TP_GPIO4_AO	DIO	SCP GPIO4	G35
TP_GPIO5_AO	DIO	SCP GPIO5	H33
TP_GPIO6_AO	DIO	SCP GPIO6	H32
TP_GPIO7_AO	DIO	SCP GPIO7	H34
SCP Command Signals			
SCP_VREQ_VAO	DO	SCP to PMIC normal voltage request	AT15

(1) These pins must be connected to an external pull-up 4.7 kΩ and to an external pull-up 2.2 kΩ for FS+ and HS modes.

3.6 Memory

3.6.1 DRAMC Controller (DRAMC)

3.6.1.1 Overview

The Dynamic Random-Access Memory Controller (DRAMC) supports the DRAM bus configuration of 4 channels of LPDDR4/4X 16-bit at 4,266 MHz (4,266 Mbps/per bit channel). When operating in the four-channel mode, this configuration enables a maximum DRAM bandwidth of up to 34.13 GB/s.

3.6.1.2 Features

The External Memory Interface (EMI) controller schedules requests from the masters and issues commands to the DRAM controller. It conducts flow control for the DRAM controller and masters to avoid DRAM stall or data overflow/underflow, minimizing the latency of the processor path to enhance the performance and increase DRAM efficiency. Furthermore, it informs clock control to gate the clock when there are no transactions to be processed.

The LPDDR4X DRAM controller supports the following features:

- Supports high-priority sideband signals for reducing the request latency.
- Supports LPDDR4X with dual-rank per channel.
- Maximum capacity of up to 16 GB for four channels with 16-bit data bus width
- Power-down and self-refresh for power saving.
- Automatic clock stops for power saving.
- DRAM I/O interface timing calibration for PVT variation
- Read/write commands out of the order control
- 7 shuffle frequencies
- Supports DVFS for the SoC.
- A request cannot cross page boundary.

The LPDDR4X DRAM supports a maximum shuffle capacity of up to 7 frequencies, enabling it to dynamically operate at the suitable frequency and voltage for the user's scenario, bandwidth requirement and low power control to achieve a balance between performance and power. Table 3-6 displays the Dynamic Frequency Scaling (DFS) with 7 operating frequencies in Mbps. The additional capability is programmable to fine-tune performance and production requirements.

Table 3-6 LPDDR4X DFS (Mbps)

LPDDR4X	4,266	3,200	2,400	1,866	1,600	1,200	800

3.6.1.3 Block Diagram

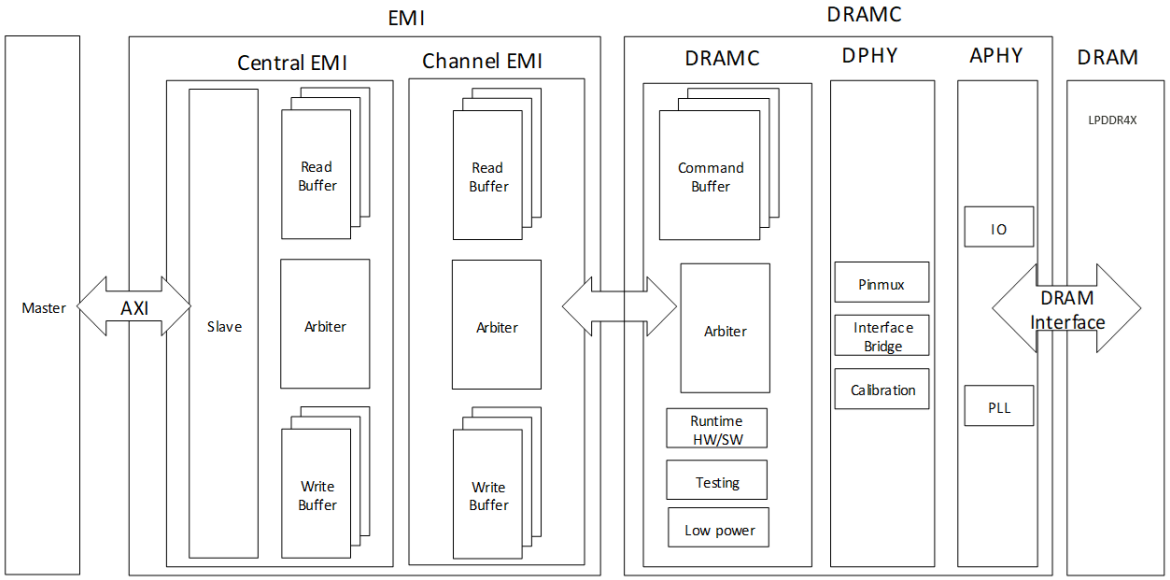


Figure 3-11 Block Diagram of Memory System

3.6.1.4 Function Description

The DRAMC transmits requests from the EMI to the DRAM protocol. Configurable registers that can be programmed by the CPU allow the DRAMC to operate in different modes.

- The requests from the EMI scheduler are pushed to the **command pool** to wait for execution in order.
- The **bus scheduler** inspects the pre-charge/active pool and command FIFO, and decides which SDRAM bus command, for example, PRECHARGE, ACTIVE, READ or WRITE, is issued to the SDRAM bus in order to maximize the bus utilization rate and reduce the response latency.
- The **timing controller** is responsible for the integrity of the SDRAM bus timing, such as pre-charge to active delay (Trp), active to command delay (Trcd) and bus turn-around time. The bus scheduler then refers to the information and chooses the next SDRAM bus command.
- The **DDR PHY unit** generates SDRAM bus commands, transmitting data and DQS to the SDRAM, and receiving data and DQS from the SDRAM.
- The **response generator** produces the response signals and data to EMI scheduler.

As Figure 3-11 shows, besides the DRAMC that consists of a lower power controller, there are also the digital physical part (DPHY) and analog physical part (APHY).

- The **low power controller** responds to DRAMC low power control, such as power domain shutdown, startup and Dynamic Voltage Frequency Scaling (DVFS).
- The **DPHY** is the middle level design covering the pin-mux and calibration operations, as well as the interface protocol between the DRAMC and APHY.
- The **APHY** is an analog high-speed DDRPHY responsible for the high-speed I/O design for the DDR Interface operation, with an integrated local PLL to generate the target frequency for the local design usage.

3.6.1.4.1 Clock

The clock source of the DRAM system (DRAMSYS) APHY is from its PLL, while the other one is from the SoC PLL, as Figure 3-12 shows. This clock structure supports the DRAMSYS in different modes, such as active, idle and DVFS, as well as dynamic clock management for the low power design.

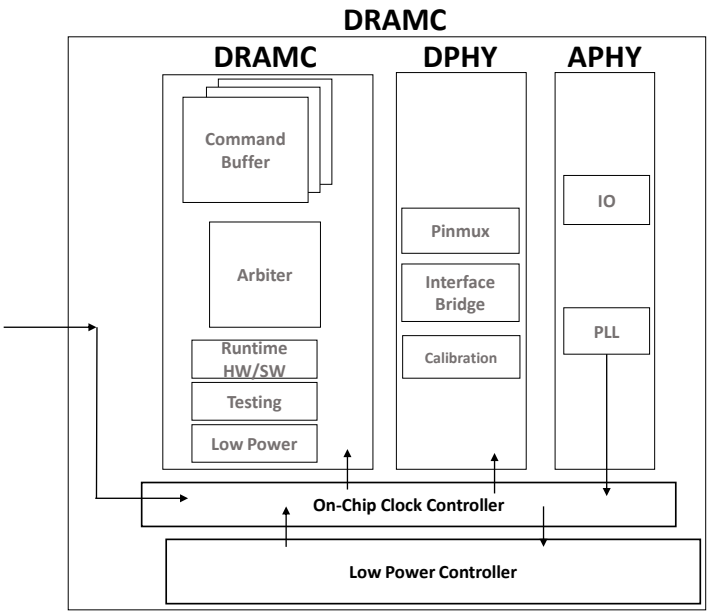


Figure 3-12 DRAMSYS Clock and Low Power Structure

3.6.1.4.2 Reset

The asynchronous reset is sourced from the SoC, while the software reset on the DRAMSYS is sourced on the internal MediaTek low power controller.

3.6.1.5 Theory of Operations

3.6.1.5.1 Write Operation

The LPDDR4-SDRAM uses an un-matched DQS-DQ path for lower power, so the DQS-strobe must arrive at the SDRAM ball prior to the DQ signal by the amount of Tdqs2DQ. The DQS-strobe output is driven Twpre before the first valid rising strobe edge. The Twpre pre-amble should be 2xtCK. The DQS strobe must be trained to arrive at the DQ pad center-aligned with the DQ-data. The DQ-data must be held for Tdivw (data input valid window) and the DQS must be periodically trained to stay centered in the Tdivw to compensate for timing changes due to temperature and voltage variation. Burst data is captured by the SDRAM on successive edges of DQS until the 16-bit or 32-bit data burst is complete. The DQS-strobe must remain active (toggling) for Twpst (WRITE post-amble) after the completion of the burst WRITE. After a burst WRITE operation, Twr must be satisfied before a PRECHARGE command to the same bank can be issued.

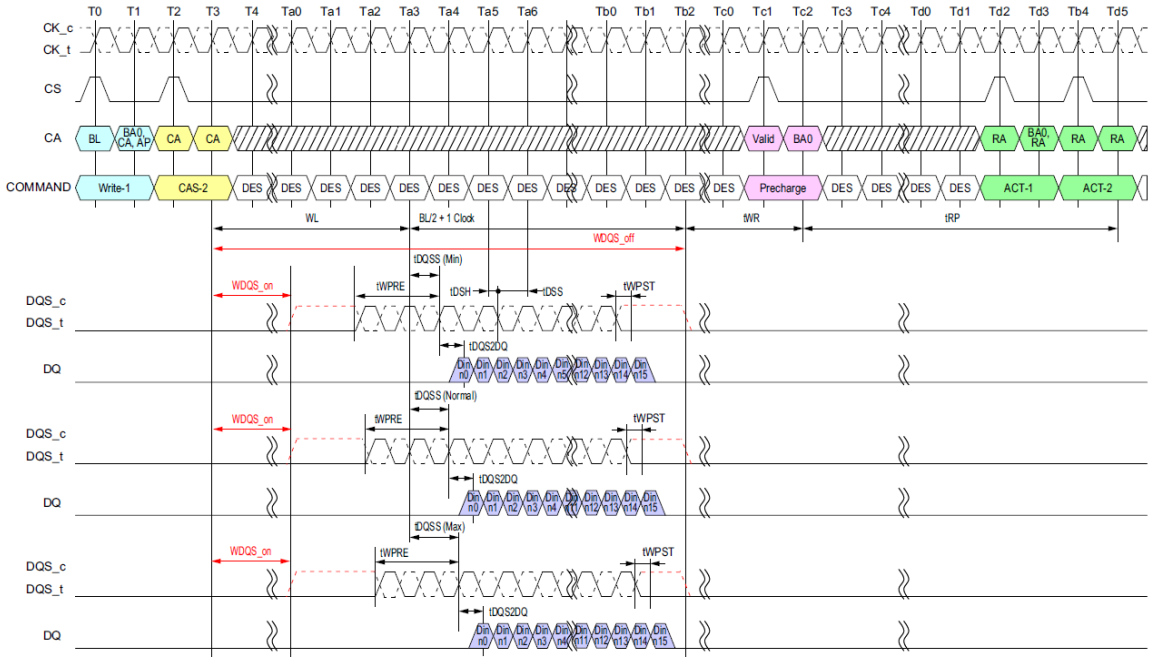


Figure 3-13 Write Operation

3.6.1.5.2 Read Operation

A burst read command is initiated with CS, and CA[5:0] asserted to the proper state at the rising edge of CK, as defined by the LPDDR4 Command Truth Table. The command address bus inputs determine the starting column address for the burst. The two low-order address bits are not transmitted on the CA bus and are implied to be “0”, so that the starting burst address is always a multiple of four, for example, 0x0, 0x4, 0x8, 0Xc. The read latency (RL) is defined from the last rising edge of the clock that completes a read command, for example, the second rising edge of the CAS-2 command, to the rising edge of the clock, from which the tDQSCK delay is measured. The first valid data is available RL * tCK + tDQSCK + tDQSQ after the rising edge of Clock that completes a read command. The data strobe output is driven tRPRE before the first valid rising strobe edge. The first data-bit of the burst is synchronized with the first valid (i.e., post-preamble) rising edge of the data strobe. Each subsequent data out appears on each DQ pin, edge-aligned with the data strobe. At the end of a burst the DQS signals are driven for another half cycle post-amble, or for a 1.5-cycle post-amble if the programmable post-amble bit is set in the mode register.

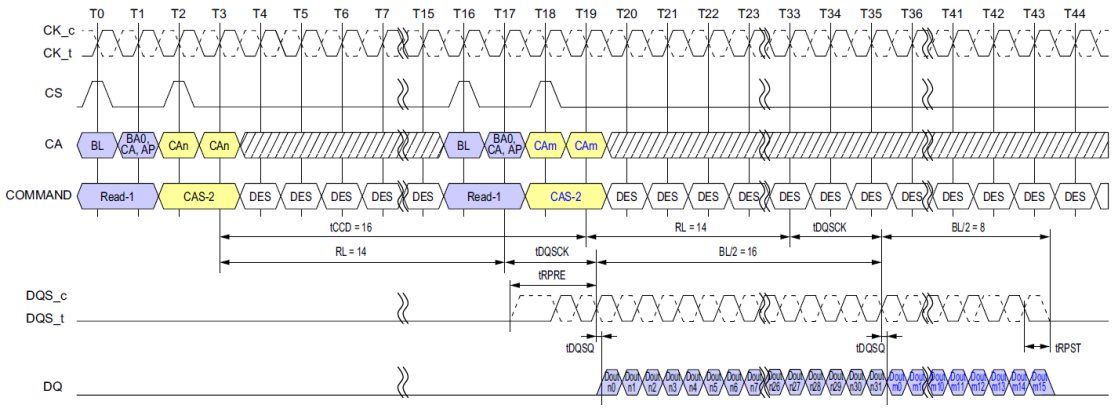


Figure 3-14 Read Operation

3.6.1.6 Power Management

The following diagram and table illustrate the power domain and voltage terms and purpose of the DRAMC. For DRAM relative voltage constraints, refer to the JEDEC standard specification.

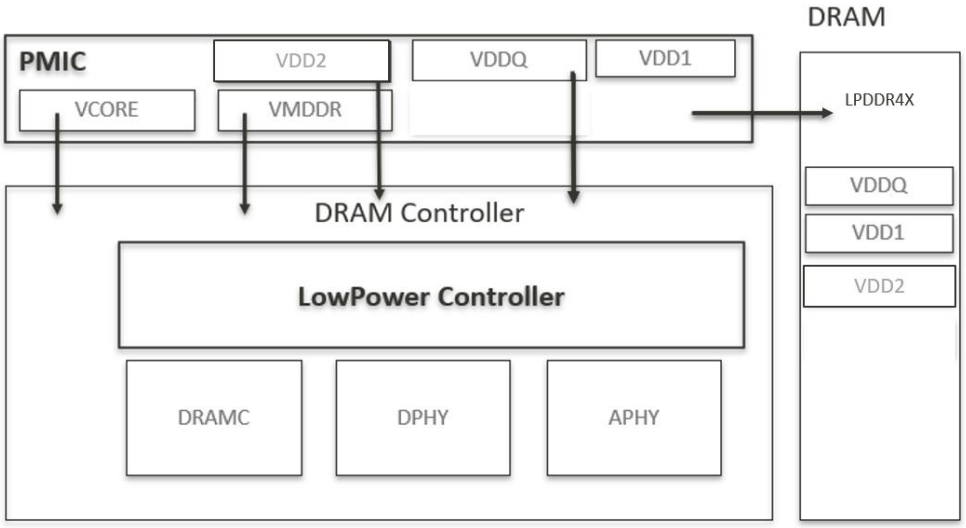


Figure 3-15 DRAM Controller Power Domain Architecture

Table 3-7 Voltage Support and Desciption

Voltage Supply	Description
VCORE	Digital circuit power domain
VMDDR	Analog DDRPHY power domain
VDD1	LPDDR4X 1.8V power rail
VDD2	LPDDR4X 1.1V power rail
VDDQ	IO buffer 0.6V power rail

3.6.1.7 LPDDR4X Signal Description

Table 3-8 DRAMC Signal Description

Signal Name	Type	Description
CK_t	Input	DRAM clock signal
CK_c	Input	DRAM clock invert signal
CKE	Input	DRAM clock enable
CS0/CS1	Input	RANK1 to RANK0 selection signal
CA[5:0]	Input	Address for all memories/CA buses for LPDDR4X
DQ[15:0]	I/O	Data bus for LPDDR4X
DMI[1:0]	Input	Data mask inversion
DQS_t[1:0]	I/O	Data strobe
DQS_c[1:0]	I/O	Data strobe invert
ZQ	Reference	Output driving calibration
RESET_n	Input	Reset

3.6.1.7.1 LPDDR4X Timing Characteristics

The EMI LPDDR4X timing characteristics are compliant with JEDEC Standard—JESD209-4B.

3.6.1.7.2 LPDDR4X Application Guidelines

Table 3-9 presents supported LPDDR4X device combinations.

Table 3-9 LPDDR4X Device Combinations

Number of Devices	Device Data Width	Mirrored	EMI Width
2	2 × 16-bit	No	64-bit

Figure 3-16 shows the schematic connections for a 64-bit interface using 2 × 2 × 16-bit devices.

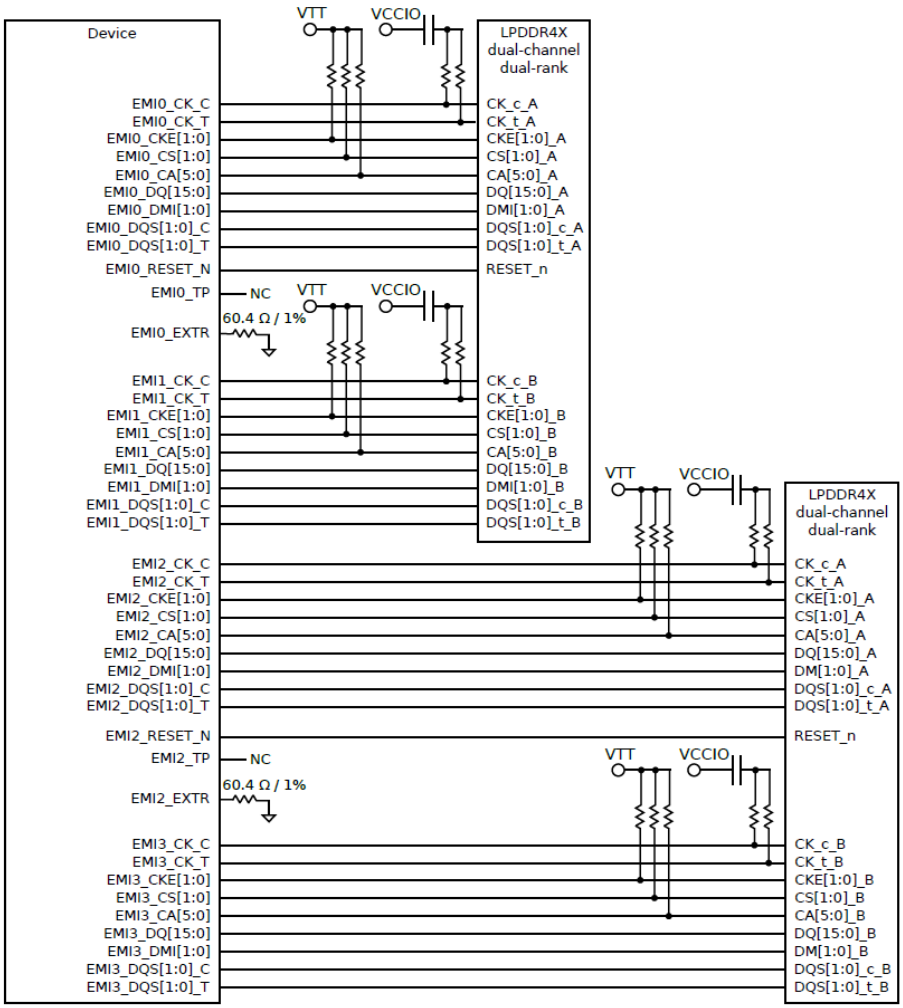


Figure 3-16 LPDDR4X Basic Schematic for 2 × 2 × 16-bit

3.6.1.8 Programming Guide

Initialize the DDRPHY.

1. Initialize the DRAMC.

2. Initialize the MediaTek low power controller.
3. Initialize the DRAM:
   - Set up the DRAM AC timing parameter.
   - Follow the DRAM specification to complete the DRAM initialization including mode-register programming.
   - Enable the command bus training.
   - Enable calibration for the DQ/DQS window.
   - Set up the refresh-rate counter.
4. Operate normally.

3.6.1.9 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.6.1.10 References

LPDDR4X spec: http://www.jedec.org/download/search/JC-42.6.pdf

3.6.1.11 Single Description

Table 3-10 presents supported LPDDR4X device combinations.

Table 3-10 EMI Signal Descriptions

Signal Name	Type	Description	Ball Location
EMI0—Calibration Resistor, Reset Output, Voltage Reference			
EMI0_EXTR(1)	AIO	EMI0 DRAM output driving calibration resistor	B3
EMI0_RESET_N	DO	EMI0 DRAM reset output	D29
EMI0_TP(2)	AIO	EMI0 DRAM voltage reference, connected to ½ AVDDQ_EMI0	H19
EMI0 Command/Address Bus—EMI0_CA[5:0]			
EMI0_CA0	DO	EMI0 DRAM command/address output 0	B18
EMI0_CA1	DO	EMI0 DRAM command/address output 1	B17
EMI0_CA2	DO	EMI0 DRAM command/address output 2	A18
EMI0_CA3	DO	EMI0 DRAM command/address output 3	C20
EMI0_CA4	DO	EMI0 DRAM command/address output 4	A20
EMI0_CA5	DIO	EMI0 DRAM command/address output 5	D19
EMI0 System Bus—Command, Chip Select, Data Mask, Data Strobe, Clock Signals			
EMI0_CK_T	DO	EMI0 DRAM clock	E17
EMI0_CK_C	DO	EMI0 DRAM clock invert	F17
EMI0_CKE0	DO	EMI0 DRAM clock enable for rank 0	G21
EMI0_CKE1	DO	EMI0 DRAM clock enable for rank 1	G20
EMI0_CS0	DO	EMI0 DRAM chip select for rank 0	F19
EMI0_CS1	DO	EMI0 DRAM chip select for rank 1	E18
EMI0_DMI0	DIO	EMI0 DRAM data mask/inversion for DQ[7:0]	A28
EMI0_DMI1	DIO	EMI0 DRAM data mask/inversion for DQ[15:8]	A22
EMI0_DQS0_T	DIO	EMI0 DRAM data strobe for DQ[7:0]	C28

Signal Name	Type	Description	Ball Location
EMI0_DQS0_C	DIO	EMI0 DRAM data strobe invert for DQ[7:0]	D28
EMI0_DQS1_T	DIO	EMI0 DRAM data strobe for DQ[15:8]	E25
EMI0_DQS1_C	DIO	EMI0 DRAM data strobe invert for DQ[15:8]	F25
EMI0 Data Bus—EMI0_DQ[15:0]			
EMI0_DQ0	DIO	EMI0 DRAM data pin 0	E27
EMI0_DQ1	DIO	EMI0 DRAM data pin 1	F26
EMI0_DQ2	DIO	EMI0 DRAM data pin 2	A30
EMI0_DQ3	DIO	EMI0 DRAM data pin 3	E26
EMI0_DQ4	DIO	EMI0 DRAM data pin 4	C26
EMI0_DQ5	DIO	EMI0 DRAM data pin 5	B29
EMI0_DQ6	DIO	EMI0 DRAM data pin 6	B27
EMI0_DQ7	DIO	EMI0 DRAM data pin 7	A26
EMI0_DQ8	DIO	EMI0 DRAM data pin 8	A24
EMI0_DQ9	DIO	EMI0 DRAM data pin 9	B23
EMI0_DQ10	DIO	EMI0 DRAM data pin 10	F24
EMI0_DQ11	DIO	EMI0 DRAM data pin 11	E23
EMI0_DQ12	DIO	EMI0 DRAM data pin 12	D21
EMI0_DQ13	DIO	EMI0 DRAM data pin 13	D22
EMI0_DQ14	DIO	EMI0 DRAM data pin 14	B21
EMI0_DQ15	DIO	EMI0 DRAM data pin 15	B25
EMI1 Command/Address Bus—EMI1_CA[5:0]			
EMI1_CA0	DO	EMI1 DRAM command/address output 0	E15
EMI1_CA1	DO	EMI1 DRAM command/address output 1	C16
EMI1_CA2	DO	EMI1 DRAM command/address output 2	A15
EMI1_CA3	DO	EMI1 DRAM command/address output 3	C13
EMI1_CA4	DO	EMI1 DRAM command/address output 4	A13
EMI1_CA5	DO	EMI1 DRAM command/address output 5	B15
EMI1 System Bus—Command, Chip Select, Data Mask, Data Strobe, Clock Signals			
EMI1_CK_T	DO	EMI1 DRAM clock	E16
EMI1_CK_C	DO	EMI1 DRAM clock invert	D16
EMI1_CKE0	DO	EMI1 DRAM clock enable for rank 0	D14
EMI1_CKE1	DO	EMI1 DRAM clock enable for rank 1	C14
EMI1_CS0	DO	EMI1 DRAM chip select for rank 0	F14
EMI1_CS1	DO	EMI1 DRAM chip select for rank 1	F15
EMI1_DMI0	DIO	EMI1 DRAM data mask/inversion for DQ[7:0]	A5
EMI1_DMI1	DIO	EMI1 DRAM data mask/inversion for DQ[15:8]	A11
EMI1_DQS0_T	DIO	EMI1 DRAM data strobe for DQ[7:0]	C5
EMI1_DQS0_C	DIO	EMI1 DRAM data strobe invert for DQ[7:0]	D5
EMI1_DQS1_T	DIO	EMI1 DRAM data strobe for DQ[15:8]	D9
EMI1_DQS1_C	DIO	EMI1 DRAM data strobe invert for DQ[15:8]	E9
EMI1 Data Bus—EMI1_DQ[15:0]			
EMI1_DQ0	DIO	EMI1 DRAM data pin 0	C6
EMI1_DQ1	DIO	EMI1 DRAM data pin 1	E7
EMI1_DQ2	DIO	EMI1 DRAM data pin 2	A3

Signal Name	Type	Description	Ball Location
EMI1_DQ3	DIO	EMI1 DRAM data pin 3	D8
EMI1_DQ4	DIO	EMI1 DRAM data pin 4	E8
EMI1_DQ5	DIO	EMI1 DRAM data pin 5	A4
EMI1_DQ6	DIO	EMI1 DRAM data pin 6	B6
EMI1_DQ7	DIO	EMI1 DRAM data pin 7	A7
EMI1_DQ8	DIO	EMI1 DRAM data pin 8	A9
EMI1_DQ9	DIO	EMI1 DRAM data pin 9	B10
EMI1_DQ10	DIO	EMI1 DRAM data pin 10	D10
EMI1_DQ11	DIO	EMI1 DRAM data pin 11	F10
EMI1_DQ12	DIO	EMI1 DRAM data pin 12	E12
EMI1_DQ13	DIO	EMI1 DRAM data pin 13	C12
EMI1_DQ14	DIO	EMI1 DRAM data pin 14	B12
EMI1_DQ15	DIO	EMI1 DRAM data pin 15	B8
EMI2—Calibration Resistor, Reset Output, Voltage Reference			
EMI2_EXTR(1)	AIO	EMI2 DRAM output driving calibration resistor	AJ1
EMI2_RESET_N	DO	EMI2 DRAM reset output	B2
EMI2_TP(2)	AIO	EMI2 DRAM voltage reference, connected to ½ AVDDQ_EMI0	R7
EMI2 Command/Address Bus—EMI2_CA[5:0]			
EMI2_CA0	DO	EMI2 DRAM command/address output 0	N5
EMI2_CA1	DO	EMI2 DRAM command/address output 1	P5
EMI2_CA2	DO	EMI2 DRAM command/address output 2	P4
EMI2_CA3	DO	EMI2 DRAM command/address output 3	N3
EMI2_CA4	DO	EMI2 DRAM command/address output 4	N1
EMI2_CA5	DO	EMI2 DRAM command/address output 5	P3
EMI2 System Bus—Command, Chip Select, Data Mask, Data Strobe, Clock Signals			
EMI2_CK_T	DO	EMI2 DRAM clock	R4
EMI2_CK_C	DO	EMI2 DRAM clock invert	R5
EMI2_CKE0	DO	EMI2 DRAM clock enable for rank 0	R2
EMI2_CKE1	DO	EMI2 DRAM clock enable for rank 1	R1
EMI2_CS0	DO	EMI2 DRAM chip select for rank 0	M5
EMI2_CS1	DO	EMI2 DRAM chip select for rank 1	N4
EMI2_DMI0	DIO	EMI2 DRAM data mask/inversion for DQ[7:0]	E1
EMI2_DMI1	DIO	EMI2 DRAM data mask/inversion for DQ[15:8]	L1
EMI2_DQS0_T	DIO	EMI2 DRAM data strobe for DQ[7:0]	C3
EMI2_DQS0_C	DIO	EMI2 DRAM data strobe invert for DQ[7:0]	D3
EMI2_DQS1_T	DIO	EMI2 DRAM data strobe for DQ[15:8]	H4
EMI2_DQS1_C	DIO	EMI2 DRAM data strobe invert for DQ[15:8]	H5
EMI2 Data Bus—EMI2_DQ[15:0]			
EMI2_DQ0	DIO	EMI2 DRAM data pin 0	C1
EMI2_DQ1	DIO	EMI2 DRAM data pin 1	D2
EMI2_DQ2	DIO	EMI2 DRAM data pin 2	F3
EMI2_DQ3	DIO	EMI2 DRAM data pin 3	G5
EMI2_DQ4	DIO	EMI2 DRAM data pin 4	F5

Signal Name	Type	Description	Ball Location
EMI2_DQ5	DIO	EMI2 DRAM data pin 5	E4
EMI2_DQ6	DIO	EMI2 DRAM data pin 6	F2
EMI2_DQ7	DIO	EMI2 DRAM data pin 7	G1
EMI2_DQ8	DIO	EMI2 DRAM data pin 8	J1
EMI2_DQ9	DIO	EMI2 DRAM data pin 9	K2
EMI2_DQ10	DIO	EMI2 DRAM data pin 10	K5
EMI2_DQ11	DIO	EMI2 DRAM data pin 11	K4
EMI2_DQ12	DIO	EMI2 DRAM data pin 12	M3
EMI2_DQ13	DIO	EMI2 DRAM data pin 13	L4
EMI2_DQ14	DIO	EMI2 DRAM data pin 14	M2
EMI2_DQ15	DIO	EMI2 DRAM data pin 15	H2
EMI3 Command/Address Bus—EMI3_CA[5:0]			
EMI3_CA0	DO	EMI3 DRAM command/address output 0	U6
EMI3_CA1	DO	EMI3 DRAM command/address output 1	U5
EMI3_CA2	DO	EMI3 DRAM command/address output 2	U3
EMI3_CA3	DO	EMI3 DRAM command/address output 3	W3
EMI3_CA4	DO	EMI3 DRAM command/address output 4	W1
EMI3_CA5	DO	EMI3 DRAM command/address output 5	V3
EMI3 System Bus—Command, Chip Select, Data Mask, Data Strobe, Clock Signals			
EMI3_CK_T	DO	EMI3 DRAM clock	T4
EMI3_CK_C	DO	EMI3 DRAM clock invert	T5
EMI3_CKE0	DO	EMI3 DRAM clock enable for rank 0	U2
EMI3_CKE1	DO	EMI3 DRAM clock enable for rank 1	U1
EMI3_CS0	DO	EMI3 DRAM chip select for rank 0	V5
EMI3_CS1	DO	EMI3 DRAM chip select for rank 1	V4
EMI3_DMI0	DIO	EMI3 DRAM data mask/inversion for DQ[7:0]	AG1
EMI3_DMI1	DIO	EMI3 DRAM data mask/inversion for DQ[15:8]	AA1
EMI3_DQS0_T	DIO	EMI3 DRAM data strobe for DQ[7:0]	AG4
EMI3_DQS0_C	DIO	EMI3 DRAM data strobe invert for DQ[7:0]	AG3
EMI3_DQS1_T	DIO	EMI3 DRAM data strobe for DQ[15:8]	AC4
EMI3_DQS1_C	DIO	EMI3 DRAM data strobe invert for DQ[15:8]	AC5
EMI3 Data Bus—EMI3_DQ[15:0]			
EMI3_DQ0	DIO	EMI3 DRAM data pin 0	AH2
EMI3_DQ1	DIO	EMI3 DRAM data pin 1	AH1
EMI3_DQ2	DIO	EMI3 DRAM data pin 2	AF5
EMI3_DQ3	DIO	EMI3 DRAM data pin 3	AD4
EMI3_DQ4	DIO	EMI3 DRAM data pin 4	AE5
EMI3_DQ5	DIO	EMI3 DRAM data pin 5	AF3
EMI3_DQ6	DIO	EMI3 DRAM data pin 6	AF2
EMI3_DQ7	DIO	EMI3 DRAM data pin 7	AE1
EMI3_DQ8	DIO	EMI3 DRAM data pin 8	AC1
EMI3_DQ9	DIO	EMI3 DRAM data pin 9	AB2
EMI3_DQ10	DIO	EMI3 DRAM data pin 10	AB4
EMI3_DQ11	DIO	EMI3 DRAM data pin 11	AA5

Signal Name	Type	Description	Ball Location
EMI3_DQ12	DIO	EMI3 DRAM data pin 12	Y5
EMI3_DQ13	DIO	EMI3 DRAM data pin 13	Y3
EMI3_DQ14	DIO	EMI3 DRAM data pin 14	Y2
EMI3_DQ15	DIO	DRAM data pin 15	AD2

(1) Connect this pin through an external 60.4 Ω (1%) resistor to GND.
(2) If not used, it can be left unconnected.

3.6.2 External Memory Interface (EMI)

3.6.2.1 Overview

The External Memory Interface (EMI) controller schedules requests from the masters and issues commands to DRAMC in an efficient way. The block conducts flow control for DRAMC and masters to avoid DRAMC stalling or data overflow or underflow. It also minimizes the latency of processor path to enhance the performance and tries to increase the DRAMC efficiency. The block also informs clock control to gate the clock when it does not find any transaction right now.

3.6.2.2 Features

The EMI controller receives AXI master commands and issues them to the DRAMC. It supports all AXI transaction type commands except for the fixed and cache commands. There are plenty of schedule options to schedule the command. The main features are as follows:

- Comply with the AXI standard protocol
  - Support SDRAM up to four channels
  - Maximum size for each channel is 8GB
- Rich QoS (Quality-of-Service) policy, as listed in Section 3.6.2.5.1

3.6.2.3 Block Diagram

The EMI connects the systems via eight 128-bit AXI ports and supports 4-channel DRAMC connection, and each channel supports two Rank SDRAMs. In each DRAMC, register programming is performed via APB interface to initialize SDRAM or other parameter settings.

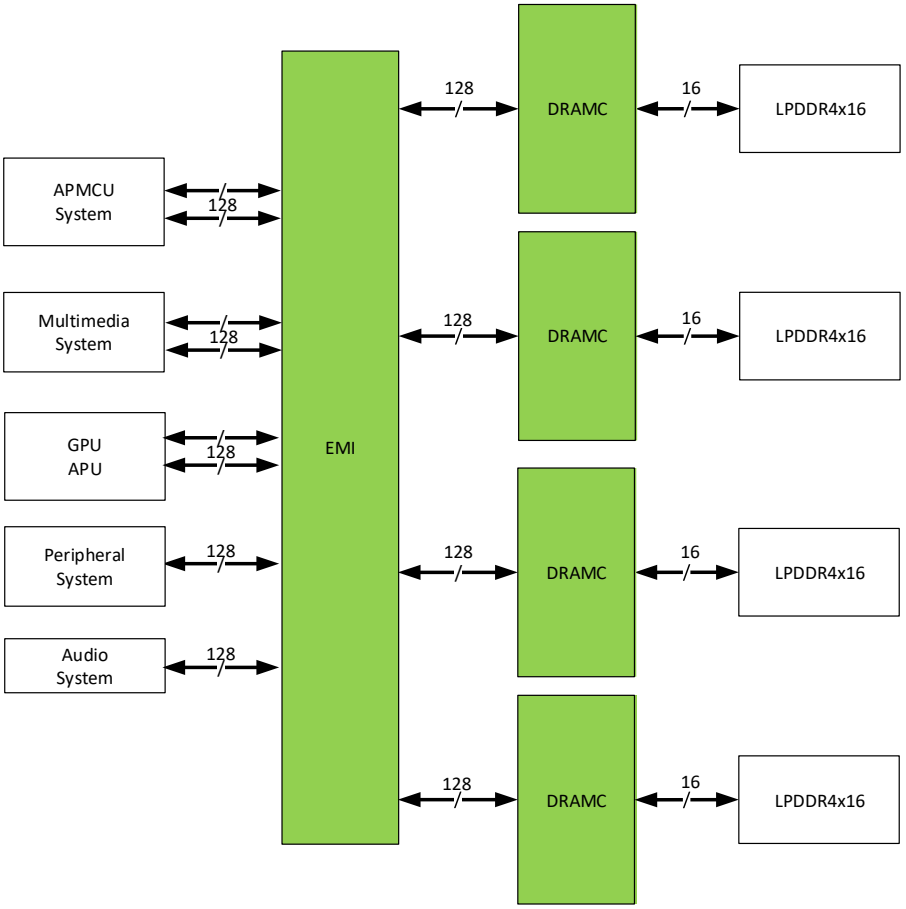


Figure 3-17 4-Channel EMI/LPDDR4 SDRAM Controller Top Connection

3.6.2.4 Function Description

The EMI is a bridge between several systems and the DRAM controller. For Cortex Application Processing Microcontroller Unit (APMCU) system, two 128-bit AXI ports are provided for the connection. For multimedia masters such as display, VENC, VDEC and camera, two 128-bit AXI ports are provided for the connection. For the GPU and APU, two 128-bit AXI ports are provided for the connection.

Besides, there is a 128-bit AXI port for connecting to the audio system and a 128-bit AXI port for connecting to the peripherals.

3.6.2.5 Theory of Operations

3.6.2.5.1 QoS Policy

To ensure that each master port can own the equal bandwidth QoS, EMI provides the following QoS Policy:

- Priority
  Each request from each master can be grouped into three levels, High, Normal and Low, and EMI serves these requests in the order of priority. Requests with ultra-signal and starvation are grouped into the High group. The requests whose allocated bandwidth is not exceeded are assigned to the Normal group. The requests whose allocated bandwidth is exceeded are categorized into the Low group.
- Starvation Prevention

To avoid request starvation, starvation counters are set for each master port. When the starvation counter reaches zero, the request is promoted to the High group for prioritized execution. Also, the read/write channel has individual starvation counter settings for either read or write latency-sensitive commands.

- Bandwidth Limiter
  EMI provides the bandwidth limiter for each master port to ensure that each master has the minimum quota for bandwidth sharing. The amount of bandwidth allocation is configurable.

3.6.2.5.2 Arbitration and Scheduling

To improve the SDRAM efficiency and minimize the latency, EMI provides several configuration options:

- Priority Code Arbitration
  EMI Scheduler follows a priority code to choose the requests with the highest priority. The priority codes depend on several domains, urgent, ultra, bandwidth limiter, page hit, and age (starvation). Age expired (with negative starvation counters) requests always have the highest priority. The domain priority can be configurable.
- Data Bus Turnaround
  Frequent changes in data direction (switching between reads and writes) would lower the SDRAM bandwidth efficiency. Therefore, scheduler would group a batch of read or write commands in a row and send them to DRAMC. The maximum group number can be configurable.
- Page Miss Prevention and Page Hit Promotion
  In order to avoid the overhead for SDRAM page miss, EMI masks the requests with page miss and promote these requests with page hit.

3.6.2.6 Programming Guide

To enable the EMI function, follow the steps below to program.

1. Program the supported channel number.
2. Program the DRAM address mapping type, such as column bit number, and bank bit number.
3. Program the latency for each AXI and set the request as high priority when the age counter expires.
4. Allocate the bandwidth requirement for each AXI port and set the bandwidth limiter value.
5. Set the bandwidth regulator for each AXI port as either soft mode or hard mode.

3.7 Storage

3.7.1 MMC (MultiMediaCard) and SD (Secure Digital) Controller (MSDC)

3.7.1.1 Overview

The MMC (MultiMediaCard) and SD (Secure Digital) Controller (MSDC) offers a high throughput data transfers while power consumption and data security between device local hosts and memory cards are taken into consideration.

The MSDC interface fully supports:

- SD 3.0 (Secure Digital) Memory Card Specification

- SDIO 3.0 (Secure Digital Input Output) Card Specification
- eMMC 5.1 (embedded MultiMediaCard) Specification

3.7.1.2 Features

The device has integrated 3 MSDC modules, MSDC0, MSDC1 and MSDC2. MSDC0 is used as the MMC™/eMMC interface, MSDC1 is used as the SD interface, and MSDC2 is used as the SD/SDIO interface.
Each MSDC module supports the following key features:

- 32-bit access on AHB for control registers
- Basic DMA and linked-list based DMA modes

The MSDC0 controller fully supports:

- 64-bit data access on AXI bus
- 1-, 4-, 8-bit data bus width for eMMC
- Backwards compatibility with legacy MMC
- High-speed Single Data Rate (SDR) mode
- High-speed Dual Data Rate (DDR) mode
- HS200 mode, SDR up to 200 MBps
- HS400 mode, DDR up to 400 MBps
- eMMC boot up mode
- Command Queue (CMDQ)
- Advanced Encryption Standard (AES)

The MSDC1 and MSDC2 controllers fully support:

- 32-bit data access on AHB
- 1-, 4-bit data bus width for SD card interface
- 1-, 4-bit data bus width for SDIO interface (MSDC2 only)
- Default Speed mode, data rate up to 12 MBps
- High-speed mode, data rate up to 25 MBps
- SDR12 mode, data rate up to 12 MBps
- SDR25 mode, data rate up to 25 MBps
- SDR50 mode, data rate up to 50 MBps
- SDR104 mode, data rate up to 100 MBps
- DDR50 mode, data rate up to 50 MBps

3.7.1.3 Block Diagram

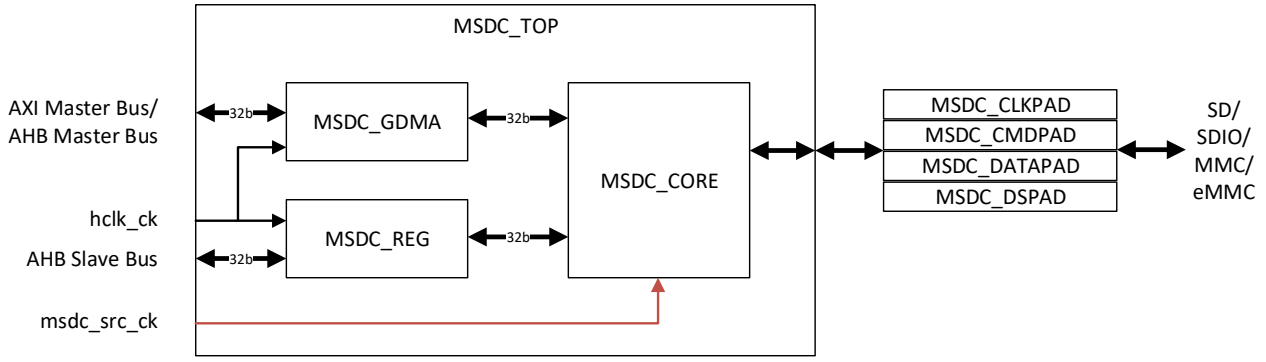


Figure 3-18 System-level Block Diagram of MSDC

3.7.1.4 Function Description

Table 3-11 MSDC Function and Address

MSDC List	Base Address	Feature
MSDC0	0x11230000	eMMC5.1
MSDC1	0x11240000	SD3.0/SDIO3.0/eMMC4.5
MSDC2	0x11250000	SDIO3.0

As Figure 3-18 illustrates, the MSDC consists of three primary components: **MSDC_GDMA, MSDC_CORE** and **MSDC_REG**.

- **MSDC_GDMA**: The DMA engine, facilitating data transfer between the MSDC and memory.
- **MSDC_CORE**: The main controller of the MSDC, managing data transfer between the host and the device.
- **MSDC_REG**: The register to configure the MSDC.

3.7.1.5 Theory of Operations

3.7.1.5.1 MSDC Read/Write

3.7.1.5.1.1 MSDC PIO Read

When MSDC_CFG.PIO_MODE is set, the MSDC operates in the PIO (Programmed Input/Output) mode and receives data from the device. The software can read data from the *PIO_RXDATA* register when MSDC_FIFOCS.RXFIFOCNT is not 0 (data buffer empty).

Upon transfer completion, an interrupt is generated. The software is responsible for clearing the interrupt bit after receiving it.

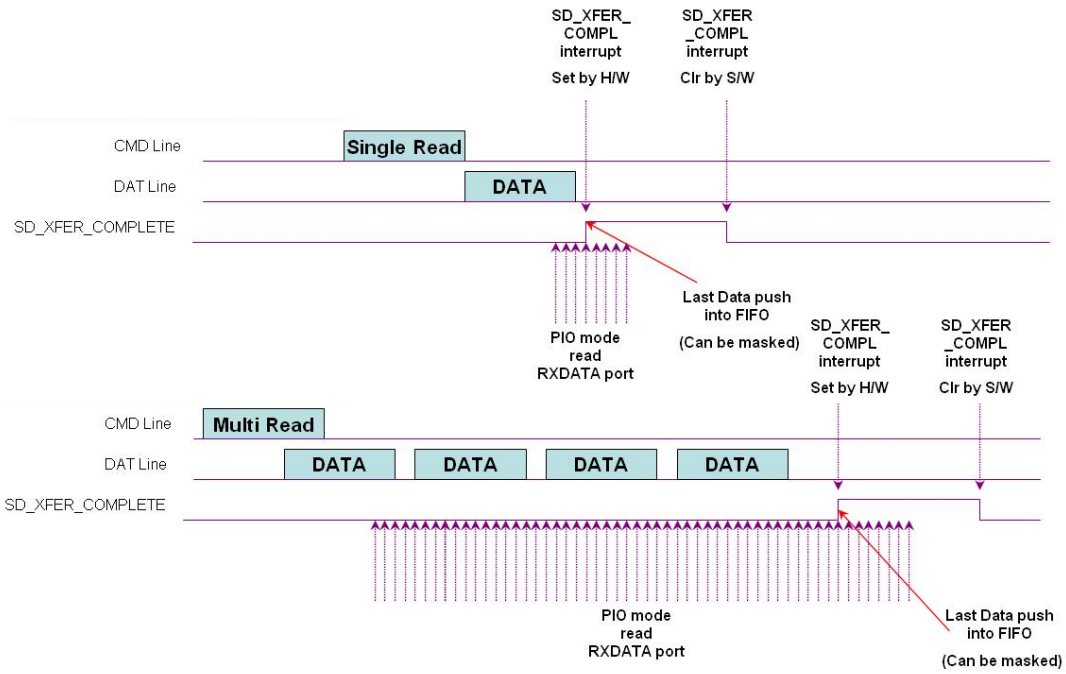


Figure 3-19 Single Block and Multi-block Read in PIO Mode

3.7.1.5.1.2 MSDC PIO Write

When MSDC_CFG.PIO_MODE is set, the MSDC operates in the PIO mode and sends data to the device. The software can write data from the *PIO_TXDATA* register when MSDC_FIFOCS.TXFIFOCNT is not 8'h80 (data buffer full).

Upon transfer completion, an interrupt is generated. The software is responsible of clearing the interrupt bit after receiving it.

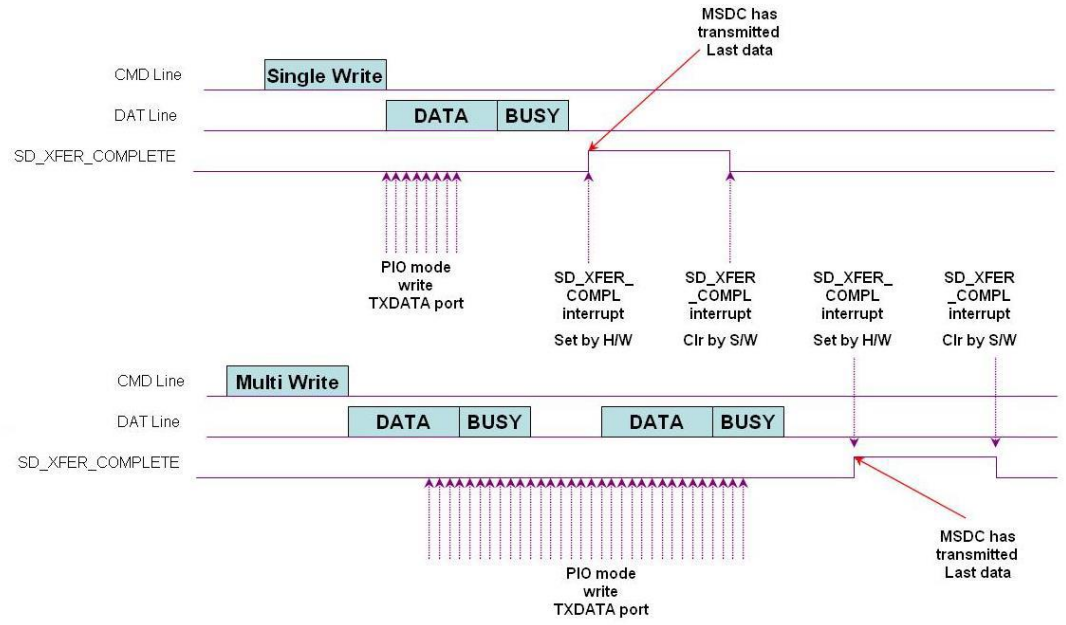


Figure 3-20 Single Block and Multi-block Write in PIO Mode

3.7.1.5.1.3 MSDC DMA Read

When MSDC_CFG.PIO_MODE is not set, the MSDC operates in the DMA mode. It receives data from the device and writes it to the target DRAM address through the MSDC_GDMA control. The software must configure the *DMA_SA* register with the start address in DRAM.

Upon transfer completion, an interrupt is generated. The software is responsible of clearing the interrupt bit after receiving it.

- **SD_XFER_COMPLETE** is set when the DMA controller has transferred all the data and the CRC has been done.
- **(DMA_CTRL.LAST_BUF = 0) DMA_DONE** is set when the DMA controller has transferred all the data set in the DMA controller.
- **(DMA_CTRL.LAST_BUF = 1) DMA_DONE** is set at the same time as SD_XFER_COMPLETE.

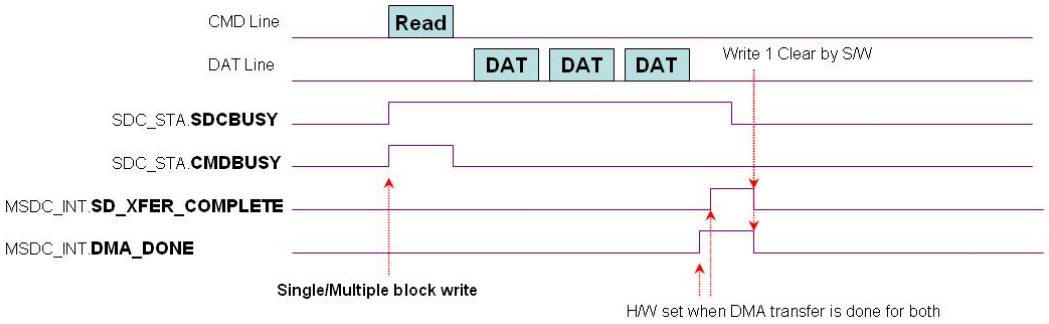


Figure 3-21 Single Block and Multi-block Read in DMA Mode

3.7.1.5.1.4 MSDC DMA Write

When MSDC_CFG.PIO_MODE is not set, the MSDC operates in the DMA mode. It receives data from DRAM through MSDC_GDMA control and writes it to the device. The software must configure the *DMA_SA* register with the start address in DRAM.

Upon transfer completion, an interrupt is generated. The software is responsible of clearing the interrupt bit after receiving it.

- **SD_XFER_COMPLETE** is set when the DMA controller has transferred all the data and the CRC has been done.
- **(DMA_CTRL.LAST_BUF = 0) DMA_DONE** is set when the DMA controller has transferred all the data set in the DMA controller.
- **(DMA_CTRL.LAST_BUF = 1) DMA_DONE** is set at the same time as SD_XFER_COMPLETE.

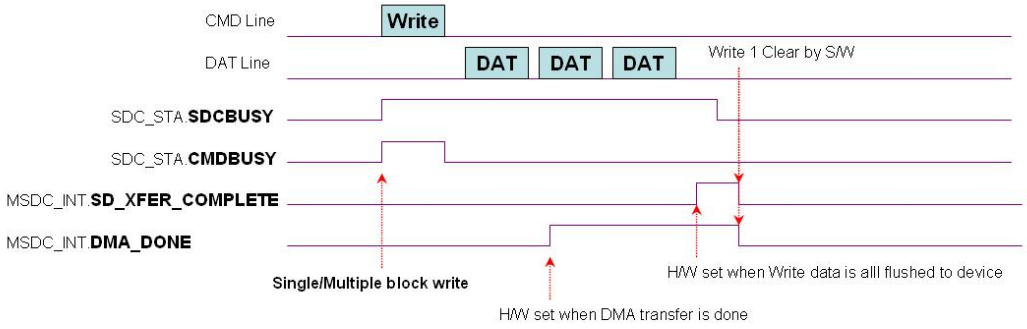


Figure 3-22 Single Block and Multi-block Write in DMA Mode

3.7.1.5.2 SD 3.0 Bus Voltage Switch

3.7.1.5.2.1 Voltage Switch Command

Figure 3-23 illustrates the definition of the voltage switch command (CMD11), which can be executed (even when the card is locked) in the ready state and does not alter the state.

A response of type R1 signifies that the card starts the voltage switch sequence. If the host detects no response, a power cycle should be executed. There are four cases, in which the card indicates no response to CMD11:

1. The card does not support voltage switching.
2. The card supports voltage switching, but ACMD41 is received with S18R = 0.
3. The card is not in the ready state.
4. The signal level is already switched to 1.8V.

In all of the above cases, CMD11 is considered an illegal command.

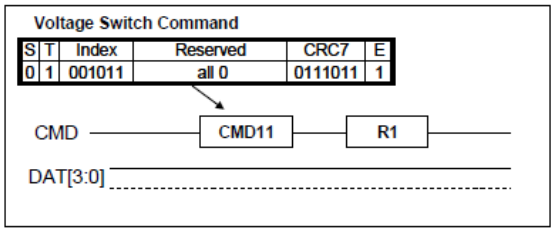


Figure 3-23 Voltage Switch Command

3.7.1.6 Programming Guide

To initiate the MSDC function, the following settings should be programmed in the prescribed manner.

3.7.1.6.1 MSDC Command Sequence

This section introduces the suggested MSDC command sequence for different scenarios.

3.7.1.6.1.1 SD Command Without Response

1. Check whether SDC_STA.SDCBUSY is 0 before starting to issue this command.

2. When this command is done, check the following bits for the status.
   - MSDC_INT.SD_CMDRDY
   - MSDC_INT.SD_CMDTO
   - MSDC_INT.SD_RESP_CRCERR

3.7.1.6.1.2 SD Command with Response

1. Check whether SDC_STA.SDCBUSY is 0 before starting to issue this command
2. When this command is done, check the following bits for the status.
   - MSDC_INT.SD_CMDRDY
   - MSDC_INT.SD_CMDTO
   - MSDC_INT.SD_RESP_CRCERR
3. The response should be in SDC_RESP0 to SDC_RESP3.

3.7.1.6.1.3 SD Command of Data Read/Write Transfer

1. Check whether SDC_STA.SDCBUSY is 0 before starting to issue this command.
2. When this command is done, check the following bits for the phase status.
   - MSDC_INT.SD_CMDRDY
   - SD_CMDTO
   - SD_RESP_CRCERR
3. The response should be in SDC_RESP0 to SDC_RESP3.
4. Enable DMA if needed (the DMA_CTRL register should be programmed).
5. The PIO mode enables data movement (the MSDC_FIFOCS, MSDC_RXDAT, and MSDC_TXDAT registers).
6. The PIO mode and DMA mode must not be switched for the same transfer; otherwise, the result can be unexpected.
7. Check the following for the data phase status.
   - MSDC_INT.SD_XFER_COMPLETE
   - DMA_DONE
   - SD_DATTO
   - SD_DATA_CRCERR

3.7.1.6.1.4 SD Software Check Before Issuing New Command

The software should always check SDC_STA.SDCBUSY before issuing a new command.

3.7.1.6.2 MSDC Tuning Support

The host controller incorporates a tuning algorithm to ensure stable data sampling, command response and CRC status from the device. Both command and data have a pad delay (pad_delay). The tuning flow is illustrated as follows.

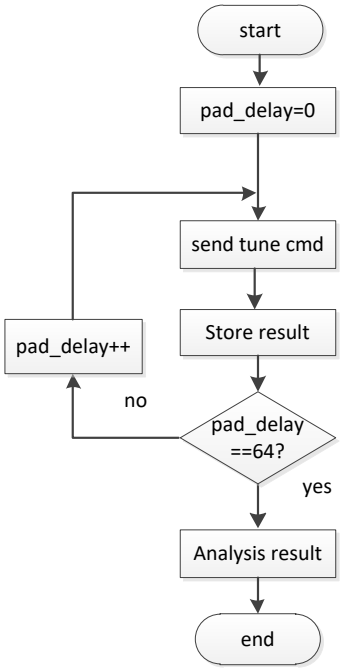


Figure 3-24 MSDC Tuning Flow

3.7.1.7 MSDC Signal Descriptions

Table 3-12 presents MSDC signal descriptions.

Table 3-12 MSDC Signal Descriptions

Signal Name	Type	Description	Ball Location
MSDC0			
MSDC0_CLK	DO	eMMC clock output	B36
MSDC0_CMD	DIO	eMMC command	C36
MSDC0_DAT0	DIO	eMMC data 0	D34
MSDC0_DAT1	DIO	eMMC data 1	D33
MSDC0_DAT2	DIO	eMMC data 2	D32
MSDC0_DAT3	DIO	eMMC data 3	C35
MSDC0_DAT4	DIO	eMMC data 4	D35
MSDC0_DAT5	DIO	eMMC data 5	E34
MSDC0_DAT6	DIO	eMMC data 6	E33
MSDC0_DAT7	DIO	eMMC data 7	E32
MSDC0_DSL	DI	eMMC data strobe input	A35
MSDC0_RSTB	DO	eMMC reset output	B35
MSDC1			
MSDC1_CLK	DO	SD card clock output	AU9
MSDC1_CMD	DIO	SD card command	AN8
MSDC1_DAT0	DIO	SD card data 0	AT6
MSDC1_DAT1	DIO	SD card data 1	AT7
MSDC1_DAT2	DIO	SD card data 2	AT8
MSDC1_DAT3	DIO	SD card data 3	AU8

Signal Name	Type	Description	Ball Location
MSDC2			
MSDC2_CLK	DO	SD card/SDIO clock output	G34
MSDC2_CLK_A	DO	SD card/SDIO clock output A	AL13
MSDC2_CMD	DIO	SD card/SDIO command	H31
MSDC2_CMD_A	DIO	SD card/SDIO command A	AP12
MSDC2_DAT0	DIO	SD card/SDIO data 0	G33
MSDC2_DAT0_A	DIO	SD card/SDIO data 0 A	AN12
MSDC2_DAT1	DIO	SD card/SDIO data 1	H33
MSDC2_DAT1_A	DIO	SD card/SDIO data 1 A	AT13
MSDC2_DAT2	DIO	SD card/SDIO data 2	G35
MSDC2_DAT2_A	DIO	SD card/SDIO data 2 A	AU14
MSDC2_DAT3	DIO	SD card/SDIO data 3	J31
MSDC2_DAT3_A	DIO	SD card/SDIO data 3 A	AP13

3.7.1.8 MSDC Signal Mapping

The communication protocol between the controller and device is implemented through an advanced 11-signal or 6-signal bus. See Table 3-13 for more details.

Table 3-13 MSDC Signal Mapping

No.	Name[3.7](1)	Type	eMMC	SD/SDHC	SDIO	Description
1	MSDC0/1/2_CLK	DO	CLK	CLK	SCLK	Clock
2	MSDC0_RSTB	DO	RCLK			Reset output
3	MSDC0/1/2_DAT0	DIO	DAT0	DAT0	DAT0	Serial data line bit 0
4	MSDC0/1/2_DAT1	DIO	DAT1	DAT1	DAT1	Serial data line bit 1
5	MSDC0/1/2_DAT2	DIO	DAT2	DAT2	DAT2	Serial data line bit 2
6	MSDC0/1/2_DAT3	DIO	DAT3	DAT3	DAT3	Serial data line bit 3
7	MSDC0_DAT4	DIO	DAT4			Serial data line bit 4
8	MSDC0_DAT5	DIO	DAT5			Serial data line bit 5
9	MSDC0_DAT6	DIO	DAT6			Serial data line bit 6
10	MSDC0_DAT7	DIO	DAT7			Serial data line bit 7
11	MSDC0/1/2_CMD	DIO	CMD	CMD	BS	Command/bus state
12	SD_WP(2)	I		WP		Write protection
13	SD_INS(2)	I	VSS2	VSS2	INS	Card insertion

(1) All embedded pull-up and pull-down resistors can be disabled by programming the corresponding control registers if optimal pull-up or pull-down resistors are required on the system board.

(2) SD_WP and SD_INS signals are not provided by the MSDC controller. These functions can be accomplished using GPIO pins, if needed.

3.7.1.9 MSDC Timing Characteristics

Table 3-14 and Figure 3-25 present the MSDC timing characteristics in Default Speed mode.

Table 3-14 MSDC Timing Characteristics (Default Speed mode)

No	Symbol	Parameter	Min	Max	Unit
Clock CLK (CLK rise and fall times are measured by min V_{IH} and max V_{IL}); C_{CARD} ≤ 10 pF					
DS1	f_{OP}	Operating frequency data transfer mode	0	25	MHz
	f_{OP_ID}	Operating frequency identification mode	100	400	kHz
DS2	$t_{w\_CLK\_L}$	Pulse duration, CLK low	10		ns
DS3	$t_{w\_CLK\_H}$	Pulse duration, CLK high	10		ns
DS4	t_{RISE_CLK}	Rise time, CLK		10	ns
DS5	t_{FALL_CLK}	Fall time, CLK		10	ns
Input DAT/CMD (referenced to CLK); C_{CARD} ≤ 10 pF					
DS6	$t_{su\_DAT/CMD}$	Setup time, DAT/CMD input	5		ns
DS7	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD input	5		ns
Output DAT/CMD (referenced to CLK); C_L ≤ 40 pF					
DS8	$t_{d\_DAT/CMD}$	Delay time, DAT/CMD output during data transfer mode	0	14	ns
DS9	t_{d_DAT/CMD_ID}	Delay time, DAT/CMD output during identification mode	0	50	ns

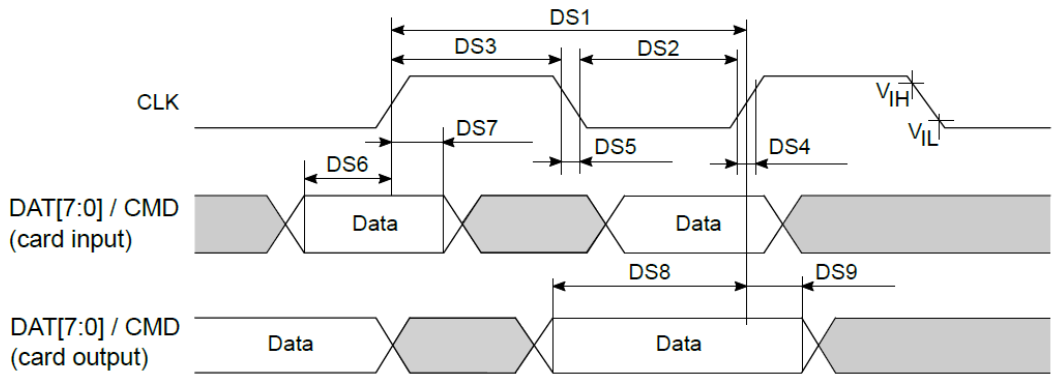


Figure 3-25 MSDC Timing Diagram (Default Speed mode)

Table 3-15 and Figure 3-26 present the MSDC timing characteristics in High Speed mode.

Table 3-15 MSDC Timing Characteristics (High Speed mode)

No	Parameter			Min	Max	Unit
Clock CLK (CLK rise and fall times are measured by min V_{IH} and max V_{IL}); C_{CARD} ≤ 10 pF						
HS1	f_{OP}	Operating frequency data transfer mode		0	50	MHz
HS2	$t_{w\_CLK\_L}$	Pulse duration, CLK low		7		ns
HS3	$t_{w\_CLK\_H}$	Pulse duration, CLK high		7		ns
HS4	t_{RISE_CLK}	Rise time, CLK			3	ns
HS5	t_{FALL_CLK}	Fall time, CLK			3	ns
Input DAT/CMD (referenced to CLK); C_{CARD} ≤ 10 pF						
HS6	$t_{su\_DAT/CMD}$	Setup time, DAT/CMD input		6		ns
HS7	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD input		2		ns
Output DAT/CMD (referenced to CLK)						
HS8	$t_{d\_DAT/CMD}$	Delay time, DAT/CMD output[1]	CL ≤ 40 pF		14	ns
HS9	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD output[1]	CL ≥ 15 pF	2.5		ns
	C_L	Total system capacitance for each line			40	pF

(1) Valid during data transfer mode.

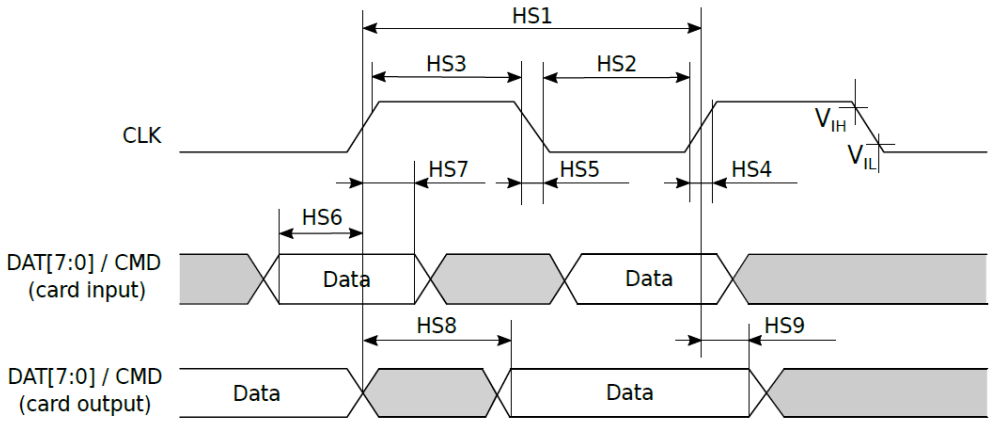


Figure 3-26 MSDC Timing Diagram (High Speed mode)

Table 3-16 and Figure 3-27 present the MSDC timing characteristics in SDR12, SDR25, SDR50, and SDR104 modes.

Table 3-16 MSDC Timing Characteristics (SDR12/SDR25/SDR50/SDR104 modes)

No.	Parameter		Min	Max	Unit
CLK output from host					
SDR121	t_C	Cycle time, CLK for SDR12		40	ns
		Cycle time, CLK for SDR25		20	ns
		Cycle time, CLK for SDR50		10	ns
		Cycle time, CLK for SDR104		4.8(4)	ns
SDR122	$t_{w\_CLK\_L}$	Pulse duration, CLK low	10		ns
SDR123	$t_{w\_CLK\_H}$	Pulse duration, CLK high	10		ns
	D	Duty Cycle, CLK	30	70	%
SDR124	t_{RISE_CLK}	Rise time, CLK		0.2 × SDR121(1)	ns
SDR125	t_{FALL_CLK}	Fall time, CLK		0.2 × SDR121(1)	ns
Host DAT/CMD input (referenced to CLK), V_{CT} = 0.975 V					
SDR126	$t_{su\_DAT/CMD}$	Setup time, DAT/CMD input for SDR50, C_{CARD} = 10 pF	3		ns
		Setup time, DAT/CMD input for SDR104, C_{CARD} = 10 pF	1.4		ns
SDR127	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD input for SDR50, C_{CARD} = 5 pF	0.8		ns
		Hold time, DAT/CMD input for SDR104, C_{CARD} = 5 pF	0.8		ns
Host DAT/CMD output (referenced to CLK)					
SDR128	$t_{d\_DAT/CMD}$	Delay time, DAT/CMD output for SDR12/SDR25, $t_C \geq 20.0$ ns, C_L = 40 pF, using driver type B		14	ns
		Delay time, DAT/CMD output for SDR50, $t_C \geq 10.0$ ns, C_L = 30 pF, using driver type B		7.5	ns
		Delay time, DAT/CMD output for SDR104	0	2	UI(2)
	$\Delta t_{d\_DAT/CMD}$	Delay variation due to temperature change after tuning for SDR104	-350	+1550	ps
SDR129	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD output for SDR12/SDR25/SDR50, C_L = 15 pF	1.5		ns

No.	Parameter		Min	Max	Unit
SDR1210	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD output for SDR104	0.6(3)		UI(2)

(1) $t_{RISE\_CLK}/t_{FALL\_CLK}$ < 0.96 ns (max) at 208 MHz, C_{CARD} = 10 pF; $t_{RISE\_CLK}/t_{FALL\_CLK}$ < 2 ns (max) at 100 MHz, C_{CARD} = 10 pF. The absolute maximum value of t_{RISE_CLK} and t_{FALL_CLK} is 10 ns regardless of the clock frequency.

(2) Unit Interval (UI) is one bit nominal time. For example, UI = 5 ns at 200 MHz.

(3) $t_{h\_DAT/CMD}$ = 2.88 ns at 208 MHz

(4) Maximum 208 MHz, V_{CT} = 0.975 V

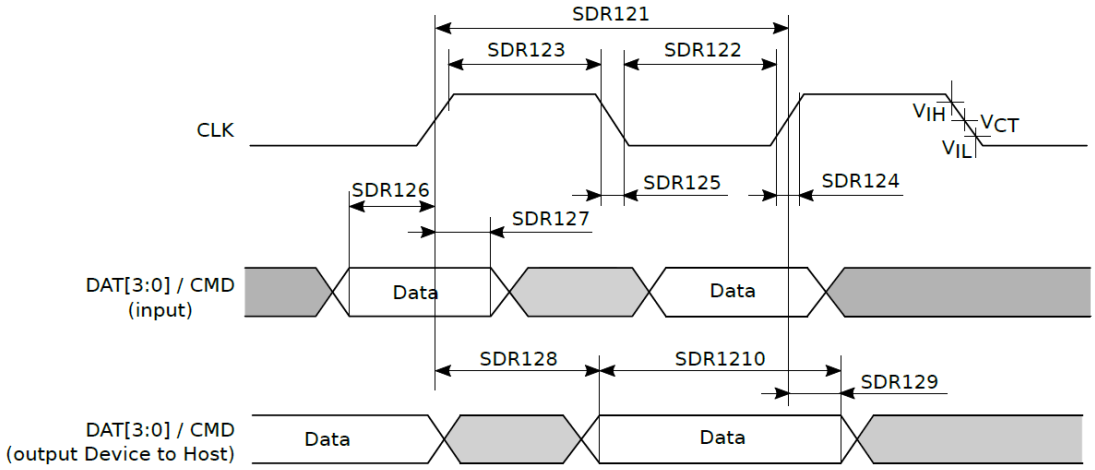


Figure 3-27 MSDC Timing Diagram (SDR12/SDR25/SDR50/SDR104 modes)

Table 3-17 and Figure 3-28 present the MSDC timing characteristics in DDR50 mode.

Table 3-17 MSDC Timing Characteristics (DDR50 mode)

No	Parameter			Min	Max	Unit
Input DAT/CMD (referenced to CLK rising and falling edge/rising edge); C_{CARD} ≤ 10 pF						
DDR503	t_{su_CMD}	Setup time, CMD input		6		ns
	t_{su_DAT}	Setup time, DAT input		3		ns
DDR504	t_{h_CMD}	Hold time, CMD input		0.8		ns
	t_{h_DAT}	Hold time, DAT input		0.8		ns
Output DAT/CMD (referenced to CLK rising and falling edge/rising edge)						
DDR505	t_{d_CMD}	Delay time, CMD output(1)	CL ≤ 30 pF		13.7	ns
	t_{d_DAT}	Delay time, DAT output(1)	CL ≤ 25 pF		7	ns
DDR506	t_{h_CMD}	Hold time, CMD output	CL ≥ 15 pF	1.5		ns
	t_{h_DAT}	Hold time, DAT output	CL ≥ 15 pF	1.5		ns

(1) Valid during data transfer mode.

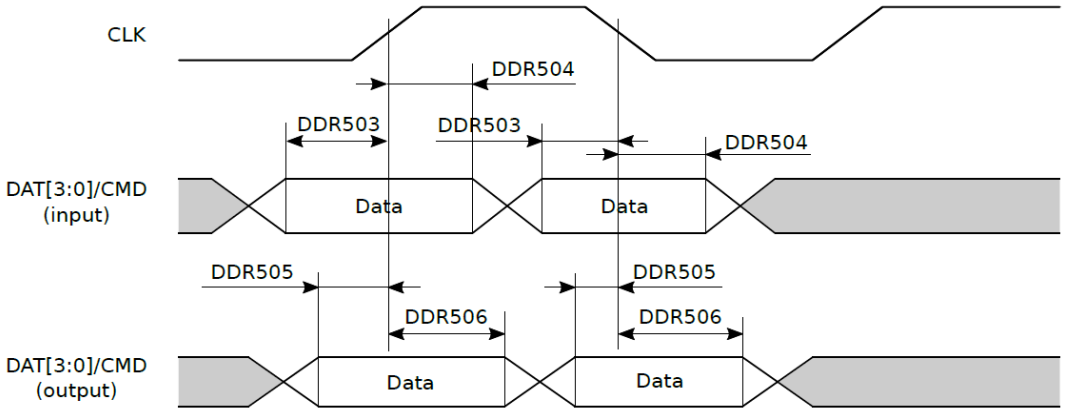


Figure 3-28 MSDC Timing Diagram (DDR50 mode)

Table 3-18 and Figure 3-29 present the MSDC timing characteristics in HS200 mode.

Table 3-18 MSDC Timing Characteristics (HS200 mode)

No	Parameter		Min	Max	Unit
Clock CLK					
HS2001	t_C	Cycle time, CLK	5		ns
HS2002	t_{RISE_CLK}	Rise time, CLK (CDevice ≤ 6 pF)		1(4)	ns
HS2003	t_{FALL_CLK}	Fall time, CLK (CDevice ≤ 6 pF)		1(4)	ns
	D	Duty Cycle, CLK	30	70	%
Input DAT/CMD; C_{Device} ≤ 6 pF					
HS2005	$t_{su\_DAT/CMD}$	Setup time, DAT/CMD input	1.4		ns
HS2006	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD input	0.8		ns
Output DAT/CMD					
HS2007	$t_{d\_DAT/CMD}$	Delay time, DAT/CMD output	0	2	UI(1)
	$\Delta t_{d\_DAT/CMD}$	Delay variation due to temperature change after tuning(2)	-350 (ΔT=-20°C)	1550 (ΔT=90°C)	ps
HS2008	$t_{h\_DAT/CMD}$	Hold time, DAT/CMD output	0.575(3)		UI(1)

(1) Unit Interval (UI) is one bit nominal time. For example, UI= 5 ns at 200 MHz.

(2) Total allowable shift of output valid window ($t_{h\_DAT/CMD}$) from last system tuning procedure $\Delta t_{d\_DAT/CMD}$ is 2600 ps for ΔT from -25 °C to 125 °C during operation.

(3) The minimum value is equal to 2.88 ns at 208 MHz.

(4) The absolute maximum value of t_{RISE_CLK} and t_{FALL_CLK} is 10 ns regardless of the clock frequency.

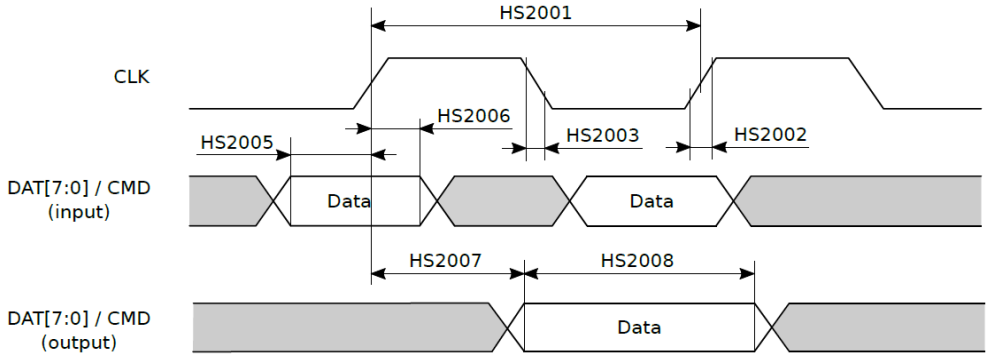


Figure 3-29 MSDC Timing Diagram (HS200 mode)

Table 3-19, Figure 3-30 and Figure 3-31 present the MSDC timing characteristics in HS400 mode.

Table 3-19 MSDC Timing Characteristics (HS400 mode)

No	Symbol	Parameter	Min	Max	Unit
Input CLK					
HS4001	t_{C_CLK}	Cycle time, CLK (with respect to V_T)	5		ns
	SR	Slew rate, with respect to V_{IH}/V_{IL}	1.125		V/ns
HS4002	t_{ck_dd}	Duty cycle distortion(1)	0	0.3	ns
HS4003	t_{W_CLK}	Pulse duration, CLK (with respect to V_T)	2.2		ns
Input DAT (referenced to CLK); with respect to V_{IH}/V_{IL}; (C_{Device} ≤ 6 pF)					
HS4004	t_{su_DAT}	Setup time, DAT input	0.4		ns
HS4005	t_{h_DAT}	Hold time, DAT input	0.4		ns
	SR	Slew rate	1.125		V/ns
Data Strobe					
HS4006	t_{C_CLK}	Cycle time, CLK (with respect to V_T)	5		ns
	SR	Slew rate (with respect to V_{OH}/V_{OL} and HS400 reference load)	1.125		V/ns
HS4007	t_{ds_dd}	Duty cycle distortion(2)	0	0.2	ns
HS4008	t_{W_CLK}	Pulse duration, CLK (with respect to V_T)	2		ns
	t_{RPRE}	Read preamble	0.4		t_{C_CLK}
	t_{RPST}	Read post-amble	0.4		t_{C_CLK}
Input DAT (referenced to Data Strobe); with respect to V_{OH}/V_{OL} and HS400 reference load					
HS4009	t_{RQ}	Output skew		0.4	ns
HS4010	t_{RQH}	Output hold skew		0.4	ns
	SR	Slew rate	1.125		V/ns

(1) Allowable deviation from an ideal 50% duty cycle. With respect to V_T. Includes jitter and phase noise.

(2) Allowable deviation from the input CLK duty cycle distortion (t_{ck_dd}). With respect to V_T. Includes jitter and phase noise.

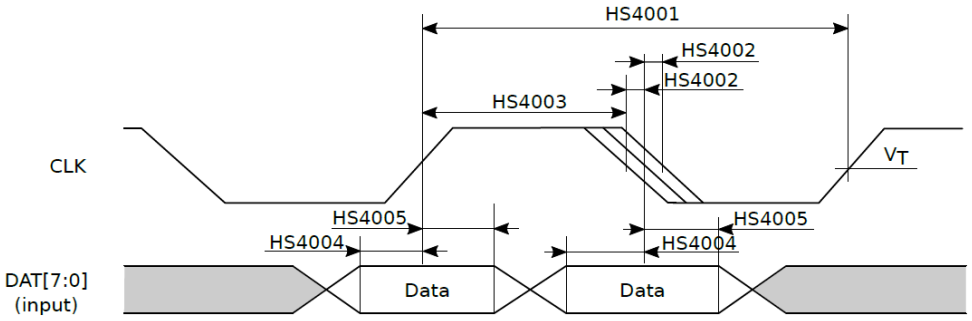


Figure 3-30 MSDC Timing Diagram (HS400 Input Mode)

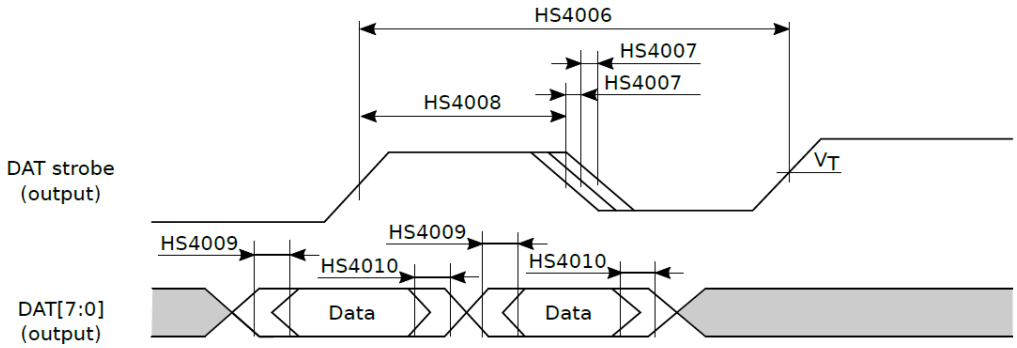


Figure 3-31 MSDC Timing Diagram (HS400 Output Mode)

3.7.2 Universal Flash Storage (UFS)

3.7.2.1 Overview

The Universal Flash Storage (UFS) allows user data to be stored in non-volatile mass storage memory devices. The characteristics of UFS include low power consumption, high data throughput, low electromagnetic interference and optimization for mass memory. Based on JEDEC Standard Universal Flash Storage version 2.1, the MT8395 UFS host controller, composed of three main modules, namely UFSHCI, UniPro and M-PHY, can provide data rate up to 5.8 Gbps with 1-lane configuration and HS-G3B speed gear.

3.7.2.2 Features

The UFS IP has the following features:

- Support 1 lane
- Support PWM-MODE G1-G4 and HS-MODE G1-G3 data rate
- Support both HS RATE A and RATE B series settings
- UniPro test feature
- Auto-Hibernate function for power saving
- AES (Advanced Encryption Standard) encryption and decryption engine for both XTS and CBC-ESSIV mode.
  - XTS mode with the key sizes of 128/256 bits
  - CBC-ESSIV with the key sizes of 128/192/256 bits
- 32 configurable slots for AES
- CPU slave interface: AHB slave design

- DMA master: AXI master design, 36-bit address width supporting 64 G memory space
- M-PHY supports 8b/10b line coding
- M-PHY TX supports large amplitude only

3.7.2.3 UFS Signal Descriptions

Table 3-20 presents UFS signal descriptions.

Table 3-20 UFS Signal Descriptions

Signal Name	Type	Description	Ball Location
UFS_MPHY_SCL	DI	UFS M-PHY serial clock	AL9
UFS_MPHY_SDA	DIO	UFS M-PHY serial data	AM10
UFS_RST_N	DO	UFS reset	A34
UFS_RX0N	AI	UFS negative differential receive data lane 0	B32
UFS_RX0P	AI	UFS positive differential receive data lane 0	B33
UFS_TX0N	AO	UFS negative differential transmit data lane 0	C31
UFS_TX0P	AO	UFS positive differential transmit data lane 0	C30
UFS_PLL_CKREF	DI	26 MHz clock input for UFS	G29
UFS_REFCK_OUT	DO	UFS reference clock output	F28

3.7.2.4 Reference

- JEDEC Standard Universal Flash Storage Version 2.1
- JEDEC UFSHCI Version 2.1
- MIPI UniPro Version 1.6
- MIPI M-PHY Version 3.1
- AMBA AXI Specification Version 2.0

3.7.2.5 Block Diagram

Figure 3-32 shows the block diagram of UFS controller. The UFS host controller is composed of three main modules: UFSHCI, UniPro and M-PHY.

M-PHY is a high-bandwidth and power-efficient physical layer, with a low-pin count serial interface. M-PHY consists of analog and digital parts. See PHYA and PHYD in the block diagram.

UniPro is the middle layer between application layer and physical layer. It defines data format, flow control, error handling and power state management.

UFSHCI is the Host Controller Interface (HCI) for UFS. The objective of UFSHCI is to provide a uniform interface method of accessing the UFS hardware capabilities, so that a standard/common driver can be provided for the host controller. In order to support cryptographic operations, a secure engine is included in UFSHCI.

MT8395 puts UniPro and M-PHY in AO domain to keep the link startup information so that when the system enters power saving mode, it still can keep the link in hibernate state. The UFS link can quickly come back after the system exits power saving state. The link startup sequence can be omitted and the UFS storage can be ready for access within a short period. It is especially beneficial for the system that needs quick response time to exit the power saving mode.

The UFS controller uses M-PHY serial interface to connect with UFS device. The interface between M-PHY and UniPro is called Reference M-PHY Module Interface (RMMI). The data interface between UniPro and UFSHCI is called Cport interface. The control interface between UniPro and UFSHCI is called Device Management Entity (DME) interface. UFSHCI uses AXI bus protocol to connect to system bus and uses AHB protocol for CPU to visit the host controller.

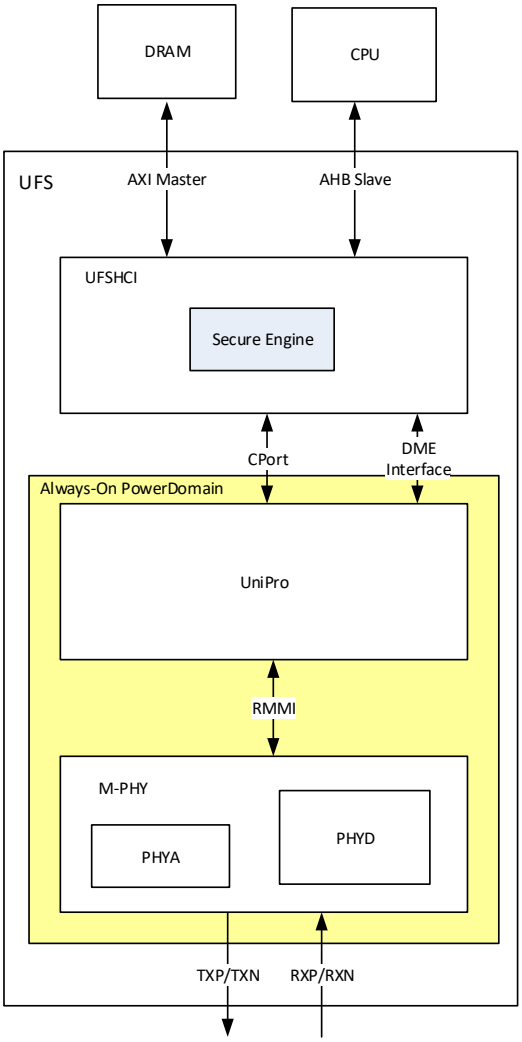


Figure 3-32 UFS Controller Block Diagram

3.7.2.6 Function Description

3.7.2.6.1 UFSHCI Layer

- UFSHCI is the Host Controller Interface (HCI) for UFS.
- The objective of UFSHCI is to provide a uniform interface for the application layer to access the UFS hardware capabilities, so that a standard/common driver can be provided for the host controller.
- UFSHCI uses AHB protocol for CPU to visit the host controller.

- AXI Master Interface is the bridge for data R/W and descriptor (UTRD, PRD, UPIU) fetch.
- AES is a powerful engine for data encryption and decryption, which is composed of multiple encrypted cores and multiple decrypted cores.
- The data interface between UniPro and UFSHCI is called Cport interface.
- The control interface between UniPro and UFSHCI is called Device Management Entity (DME) interface.

Figure 3-33 shows the block diagram of UFSHCI.

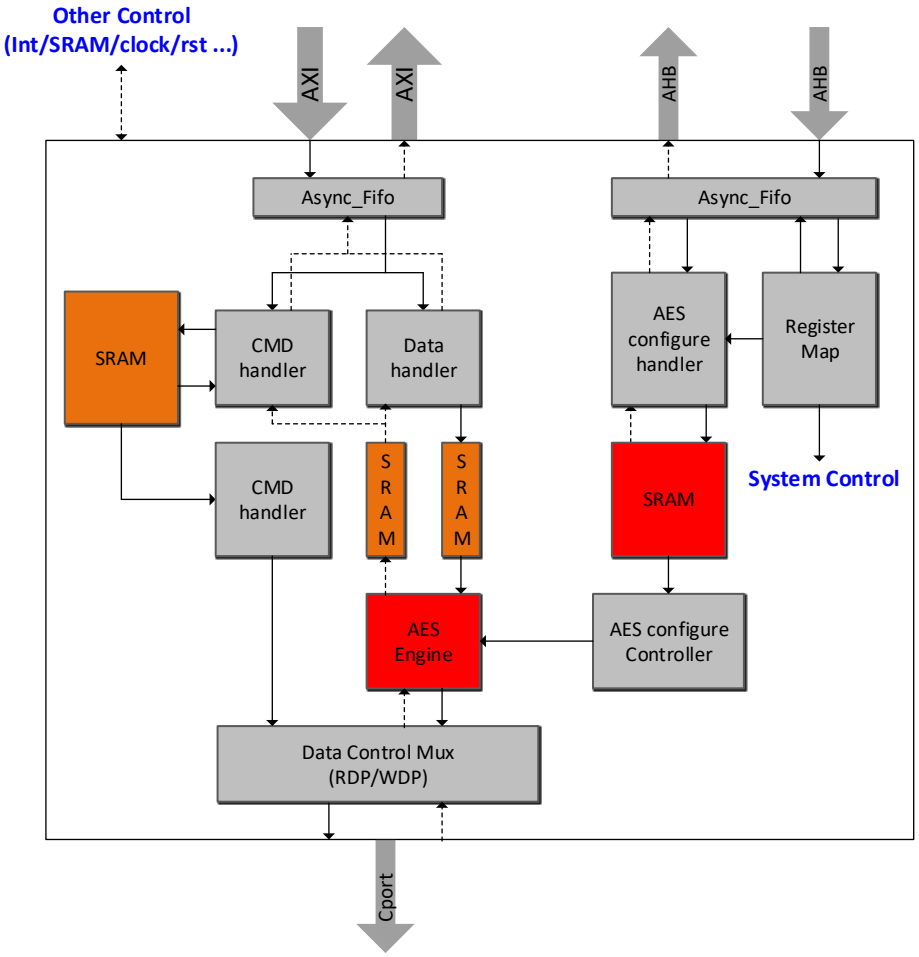


Figure 3-33 UFSHCI Block Diagram

3.7.2.6.2 AES Engine

- AES module is used to implement AES encryption and decryption algorithm. The module includes a standard APB slave interface for register access and simple valid/ready Data In/Data Out interface.
- The AES module consists of multiple encryption and decryption cores, which can improve the performance of the AES engine.

Figure 3-34 is a detailed block diagram for AES.

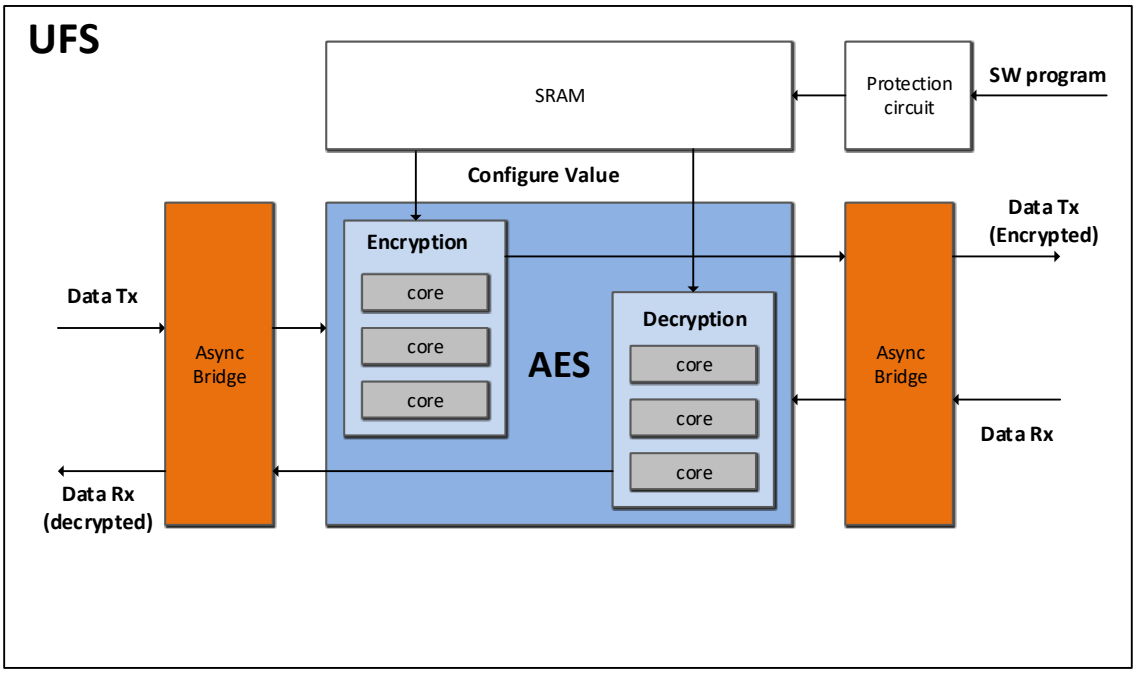


Figure 3-34 UFS AES Block Diagram

3.7.2.6.3 UniPro Layer

Figure 3-35 shows the block diagram of UniPro. Basically, it is composed of Layers 1.5, 2, 3, 4 and DME.

- L1.5: The PHY adapted layer is responsible for abstracting the details of the PHY technology, thus providing a PHY-independent interface to higher protocol layer.
- L2: The main responsibilities of the data link layer are to provide reliable Links between a transmitter and a directly attached receiver and to multiplex and arbitrate multiple types of data traffic.
- L3: The purpose of the network layer is to allow data to be routed to the proper destination in a networked environment.
- L4: The transport layer is the highest UniPro protocol layer involved in the transportation of data. It provides the data service interface called C_Port, which is used by hardware or software.
- DME: The main control block of UniPro providing attributes for the application layer to access and control.

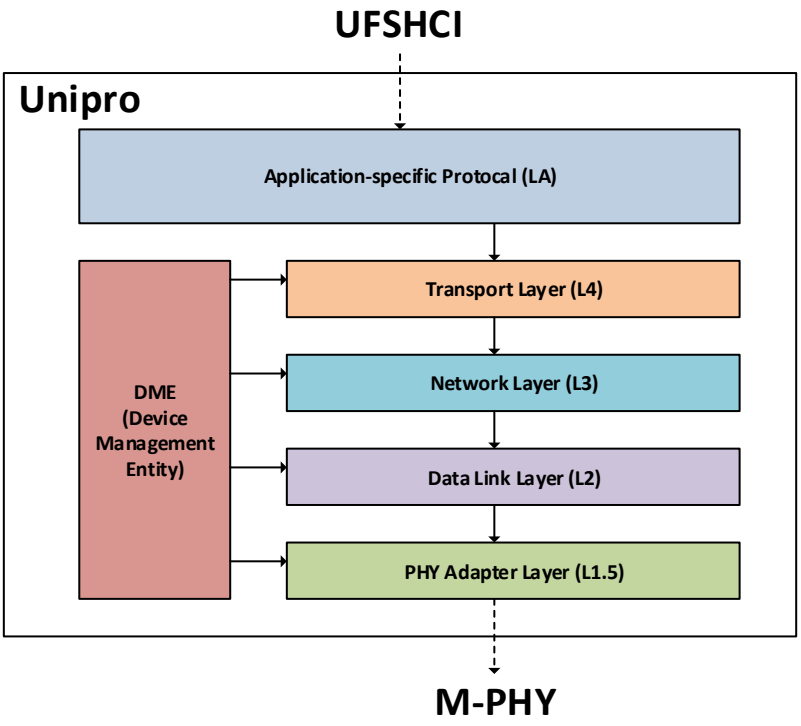


Figure 3-35 UniPro Block Diagram

3.7.2.6.4 M-PHY Layer

M-PHY is the physical layer in the UFS controller. It contains both digital and analog circuits, called PHYD and PHYA. M-PHY is a high-bandwidth and power-efficient physical layer, with a low-pin count serial interface. Figure 3-36 shows the block diagram of M-PHY. The RMMI includes control and data interface. Some MIB attributes defined in M-PHY specification are allowed to be accessed by UniPro through this control interface. Also, M-PHY provides an AHB interface for CPU to access these attributes and other internal registers, which are generally used for testing and debugging. The detailed RMMI decryption is defined in M-PHY specification.

M-PHY 3.1 specification adopts 8b/10b line coding, which is a DC-balanced coding scheme. Both PWM data transfers and HS data transfers are compliant with the 8b/10b coding rule. M-PHY also provides attributes to bypass 8b/10b coding, which allows raw data transfer from UniPro layer or from pattern generator to line. These parallel digital data is sent to PHYA and transmitted by differential output voltage signals.

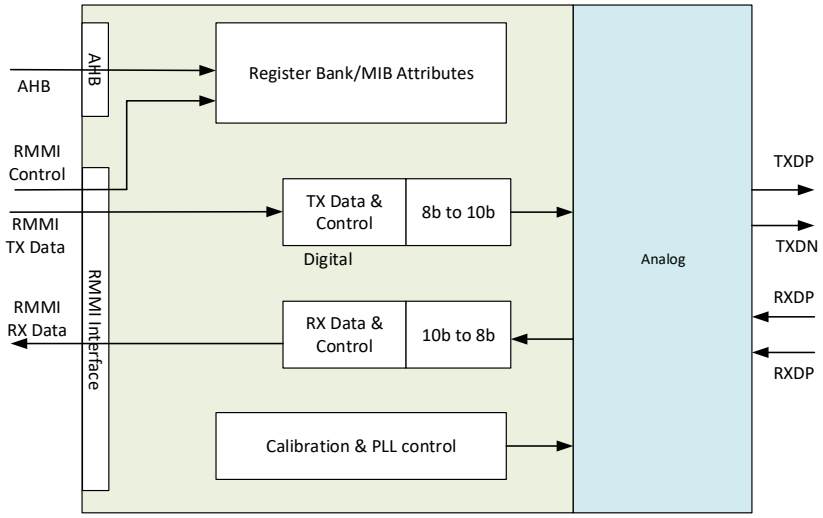


Figure 3-36 M-PHY Block Diagram

3.7.2.7 Theory of Operations

Most communications between the host software and the UFS subsystems are via system memory descriptors. These descriptors describe commands to be executed, and data transfer operations that are part of those commands. The software operation of the UFSHCI is divided into three categories: Host Controller Configuration and Control, Data Transfer Operation and Task Management. For the detailed operation, refer to the UFSHCI 2.1 specification defined by JEDEC. The UFS controller is fully compliant with this standard.

The basic operation flow of the UFS controller is that the software prepares the descriptors in Dynamic Random Access Memory (DRAM) and then sets the UFSHCI register to start command or data transfer to the UFS device. UFSHCI uses Direct Memory Access (DMA) to fetch the descriptor in DRAM, analyzes the descriptor, and then starts the transfer. The data format between the host controller and the UFS device is called UFS Protocol Information Unit (UPIU). The UPIU is sent to UniPro and wrapped up in UniPro-defined frame format, before being transferred by M-PHY using the serial interface. UFSHCI can also program UniPro and M-PHY attributes by DME interface to control different power modes of M-PHY and UniPro for high-speed transmission or low-power state during idle.

3.7.2.8 UFS 2.1 Electrical Characteristics

UFS 2.1 PHY electrical characteristics are compliant with M-PHY Specification, Revision 3.1. Refer to M-PHY specification for more detailed information.

Table 3-21 UFS 2.1 Frequency Offset Electrical Characteristics

Description	Min	Typ.	Max	Unit
Transmitter frequency offset	-2000	-	2000	ppm

Table 3-22 UFS 2.1 Transmitter and Receiver Electrical Characteristics

Description	Min	Typ.	Max	Unit
Transmitter Parameters				
Data rate series-A	1.248	2.496	4.992	Gb/s
Data rate series-B	1.456	2.912	5.824	Gb/s

Description	Min	Typ.	Max	Unit
Unit interval	171.7	-	801.2	ps
TX_{RT} differential peak to peak voltage swing(1)	-	400	-	mV
TX eye width	0.55	-	-	UI
DC single-ended TX impedance	40	-	60	Ω
TX AC common mode voltage active	-	200	-	mV
TX de-emphasis ratio	-6	-	-3.5	dB
PWM G1 transmit bit duration	-	153.8	-	ns
PWM G4 transmit bit duration	-	19.23	-	ns
PWM TX_{NT} differential peak to peak voltage swing	-	800	-	mV
Receiver Parameters				
PWM receive ratio	0.6/0.4	-	0.75/0.25	
DC differential RX impedance	80	-	110	Ω
RX_{RT} differential peak to peak voltage swing(1)	120	-	490	mV
RX AC common mode voltage active	-	200	-	mV
RX squelch exit voltage	50	-	140	mV
BER	-	-	10^{-10}	

(1) T/RX swing voltage when the differential line is terminated.

3.7.2.9 Programming Guide

All UFS host controller register interfaces are designed according to JEDEC UFSHCI 2.1. The initialization flow, power mode change sequence and data transfer operation are all available in the specification.

3.7.2.9.1 Link-Up Process Example

Step	Sequence	REG name	REG value	Address	Description
1	Set HCE enable	UFS_MMIO_ADDR_HCE	Write 'd1	0x34	Trigger link startup reset. The bit will be automatically cleared to 0 after SW program.
2	Wait for HCE ready	UFS_MMIO_ADDR_HCE	Read 'd1	0x34	Wait for HCE to become 1 again.
3	Program UIC CMD	UFS_MMIO_ADDR_UICCMD	0x16	0x90	Set UIC CMD to start link startup process.

3.7.2.9.2 Data Transfer Example

Step	Sequence	REG Name	REG Value	Address	Description
1	Prepare UTRD	-	-	-	Program the basic information for a transfer: Command type Data direction Interrupt enable Crypto enable Command Descriptor Base Address

Step	Sequence	REG Name	REG Value	Address	Description
					Response UPIU Offset/Length PRDT Offset/Length
2	Prepare Data UPIU	-	-	-	Prepare detailed transfer information.
3	Prepare Data PRD	-	-	-	Prepare the data information for a transfer: Data base address Data byte count
4	Set Interrupt Enable	UFS_MMIO_ADDR_IE	User consideration	0x24	Set the event that will trigger an interrupt.
5	Trigger DBR	UFS_MMIO_ADDR_UTRLDBR	User consideration	0x58	Trigger the DBR to start the transfer.

3.7.3 SPI NAND Flash Interface (SNFI)

3.7.3.1 Overview

The NAND flash interface (NFI) and error checking and correction (ECC) engine (in NFI mode) automatically generates ECC parity bits when NAND flash is being programmed. If you approve the way by which the NFI and ECC engine stores the parity bits in the spare area for each page, the HW_ECC mode can be used. Otherwise, you have to prepare the data, which might contain operating system information or ECC parity bits, for the spare area with certain arrangement. In the former case, the NFI and ECC engine (in NFI mode) check the parity bits when reading the NAND flash. The ECC module features BCH code, which is capable of correcting up to a 24-bit error within one sector.

3.7.3.2 Features

The device includes one SPI NAND Flash Interface (SNFI) controller.

The SNFI controller supports the following key features:

- ECC engine (BCH code acceleration allows 24-bit error correction)
- Programmable page size and spare size
- Programmable Flash Data Memory (FDM) data size and protected FDM data size
- Word/byte access through APB
- DMA for massive data transfer
- Latch sensitive interrupt (indicates the ready state for Read, Program and Erase operations)
- Programmable wait states
- Programmable command/address setup and hold times
- Programmable read enable hold time and write enable recovery time
- One chip select for SPI NAND flash parts
- Quad and Dual command modes
- Device clock, sample clock, and data skew adjustments

3.7.3.3 Block Diagram

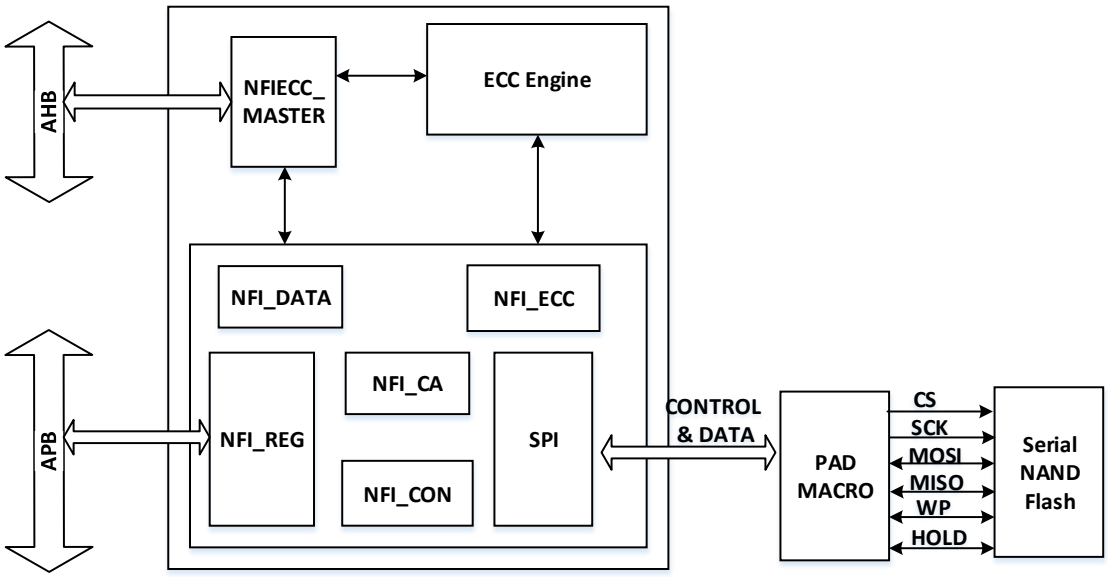


Figure 3-37 SNFI Block Diagram

NAND Flash Controller uses APB Slave Bus for accessing register configuration and data read/write, and uses AHB Master Bus for faster data read/write. It also supports interrupts, which are level active for the interrupt process. The ECC engine is used for encoding/decoding user data when needed. The NAND Flash Controller uses the standard protocol for communication with NAND devices.

3.7.3.4 Function Description

The function of NFI is to control the IC plug-in NAND device for data program and read. By means of sending control signals (may also need to occupy the DATA line to send Command, Address and Data) to NAND, the operation of the NAND can be completed, for example, Reset, Read NAND ID, Read Status, Block Erase, Page Program and Page Read. The control signals (may also need to take the DATA line to send Command, Address and Data) that meet the demand of NAND device timing diagram can be sent to NAND by configuring the specific registers of the NFI. If it can be achieved, adjust the timing of NAND according to the actual situation of the timing parameter. It is easy to handle the NAND according to user demands. The ECC engine is used for encoding/decoding user data when needed. The NAND Flash Controller uses standard protocol for communication with NAND devices.

3.7.3.5 Theory of Operations

There are four important registers that can be set, including AUTO_FMT_EN, HW_ECC_EN, READ_MODE and AHB_MODE.

- AUTO_FMT_EN can enable automatic data format for encoding or decoding operations of the hardware ECC engine. If this bit is enabled, the ECC parity from hardware ECC engine and FDM data from Register are written automatically to the spare area. If this bit is disabled, the spare area data all comes from MCU or memory.
- HW_ECC_EN can enable encoding or decoding operations of the hardware ECC engine. If this bit is enabled, the data is transferred to the ECC engine for encoding and decoding operations.
- READ_MODE can control the activity of read or write transfer.
- AHB_MODE can control the operation mode.

Table 3-23 SNFI Important Registers

AUTO_FMT_EN	HW_ECC_EN	NFI Function
0	0	Main Data, FDM data, Parity data all come from MCU or AHB. ECC Interface is off.
0	1	Main Data, FDM data, Parity data all come from MCU or AHB. ECC Interface is on.
1	0	Main Data comes from MCU or AHB, FDM data comes from Register and Parity data forces 0xff. ECC Interface is off.
1	1	Main Data comes from MCU or AHB, FDM data comes from Register and Parity data comes from ECC engine. ECC Interface is on.

Table 3-24 SNFI Important Registers

READ_MODE	AHB_MODE	NFI Function
0	0	Write transfer, PIO MODE, from MCU to NFI
0	1	Write transfer, DMA MODE, from AHB to NFI
1	0	Read transfer, PIO MODE, from NFI to MCU
1	1	Read transfer, DMA MODE, from NFI to AHB

3.7.3.6 Signal Descriptions

Table 3-25 presents SNFI signal descriptions.

Table 3-25 SNFI Signal Descriptions

Signal Name	Type	Description	Ball Location
SNFI_CLK	DO	SNFI clock	AP22
SNFI_CS	DO	SNFI chip select	AN23
SNFI_HOLD	DIO	SNFI hold	AL8
SNFI_MISO	DIO	SNFI MISO	AP23
SNFI_MOSI	DIO	SNFI MOSI	AN22
SNFI_WP	DIO	SNFI write protection	AL7

3.7.3.7 Timing Characteristics

The AC timing for output timing is shown in Table 3-26 and Figure 3-38.

Table 3-26 SNFI Output AC Timing

Parameter	Description	Min	Typ	Max	Unit
F_{ck}	SFCK frequency	26	-	104	MHz
T_{ck}	The period of SFCK	-	$1000/F_{ck}$	-	ns
t_{CSS}	Chip select active setup time related to SFCK	-	$1.5*T_{ck}$	-	ns
t_{CSH}	Chip select active hold time related to SFCK	-	$1.5*T_{ck}$ or $7.5*T_{ck}$ (1)	-	ns
t_{CLL}	Clock low pulse time	-	$0.5*T_{ck}$	-	ns

Parameter	Description	Min	Typ	Max	Unit
t_{CLH}	Clock high pulse time	-	0.5*T_{ck}	-	ns
t_{SUDAT}	Data output setup time	0.5*T_{ck}-1.5		0.5*T_{ck}-0.6	ns
t_{HDDAT}	Data output hold time	0.5*T_{ck}-1.5		0.5*T_{ck}-0.6	ns

(1) Whether t_{CSH} is 1.5*T_{ck} or 7.5*T_{ck} depends on operation modes. When in MAC mode, it is 1.5* T_{ck}, and when in auto mode, it is 7.5*T_{ck}.

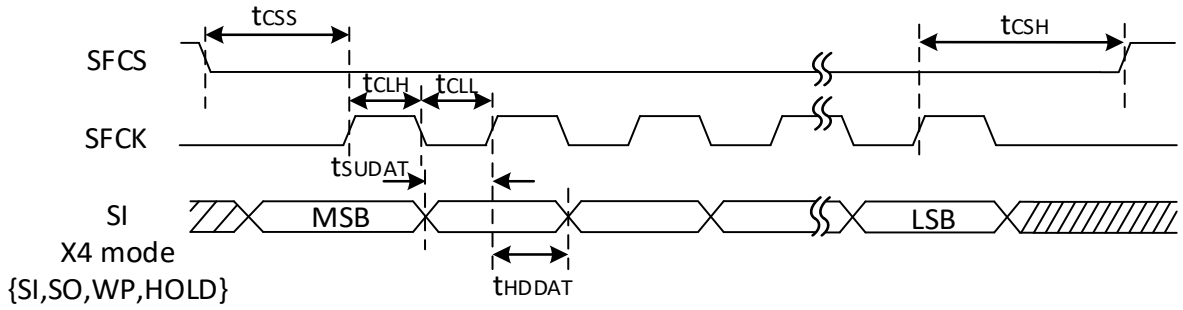


Figure 3-38 SNFI Output Timing Diagram

The AC timing for input timing is shown in Table 3-27 and Figure 3-39.

Table 3-27 SNFI Controller Input AC Timing

Parameter	Description	Min	Max	Unit
t_V	Clock LOW to output valid required time	0	12	ns
t_{HO}	Output hold required time	0	-	ns

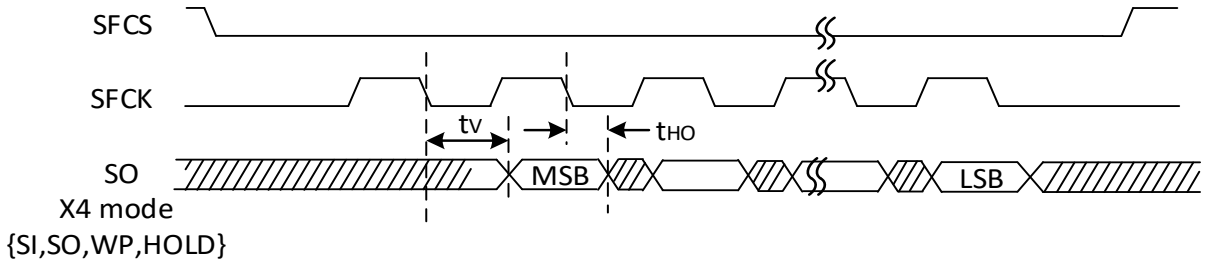


Figure 3-39 SNFI Controller Input Timing Diagram

3.7.3.8 Programming Guide

This section lists the program sequences for the SPI NAND flash operation.

- Read ID (MAC Mode)

Configuration	Memo
*CMD_SNF_MISC_CTL = 0x0400010a;	//reg_latch_ltc = 1 for 104 MHz
*CMD_SNF_MAC_CTL = 0x8;	
*SPI_GPRAM_ADDR = 0x9f;	
*CMD_SNF_MAC_OUTL = 0x2;	
*CMD_SNF_MAC_INL = 0x2;	

Configuration	Memo
*CMD_SNF_MAC_CTL = 0xc;	
//Wait for device operation finishing	
*CMD_SNF_MAC_CTL = 0x0;	//MAC_mode = 0
status = *CMD_SNF_MAC_CTL;	//Read MAC mode status
gpram_data = *SPI_GPRAM_ADDR;	//Read gpram data (ID)

- Set Feature (MAC Mode)

Configuration	Memo
*CMD_SNF_MISC_CTL = 0x0400010a;	//reg_latch_ltc = 1 for 104 MHz
*CMD_SNF_MAC_CTL = 0x8;	
*SPI_GPRAM_ADDR = 0xa01f;	//Clear BP0 to BP3 for unlocking all addresses //Based on Micro spec
*CMD_SNF_MAC_OUTL = 0x3;	
*CMD_SNF_MAC_INL = 0x0;	
*CMD_SNF_MAC_CTL = 0xc;	
//Wait for device operation finishing	
*CMD_SNF_MAC_CTL = 0x0;	//MAC_mode=0

- Write Enable (MAC Mode)

Configuration	Memo
*CMD_SNF_MAC_CTL = 0x8;	//reg_latch_ltc = 1 for 104 MHz
*SPI_GPRAM_ADDR = 0x6;	
*CMD_SNF_MAC_OUTL = 0x1;	//Clear BP0 to BP3 for unlocking all addresses //Based on Micro spec
*CMD_SNF_MAC_INL = 0x0;	
*CMD_SNF_MAC_CTL = 0xc;	
Configuration	**Memo**
// Wait for device operation finishing	
*CMD_SNF_MAC_CTL = 0x0;	

- Auto Block Erase (Auto Mode)

Configuration	Memo
*CMD_SNF_ER_CTL2 = 0x542310;	
*CMD_SNF_GF_CTL3 = 0xf0020;	
*CMD_SNF_MISC_CTL = 0x10a;	
*CMD_SNF_ER_CTL = 0xd801;	
// Wait for device operation finishing	//Keep polling status or wait for an interrupt
*CMD_SNF_ER_CTL = 0xd800;	

- Auto Program Load (Auto Mode)

Configuration	Memo
*CMD_SNF_PG_CTL1 = 0x00100206;	//Program flow command
*CMD_SNF_PG_CTL2 = 0x0;	//Program Load address

Configuration	Memo
*CMD_SNF_PG_CTL3 = 0x0;	//Program Execute address
*CMD_SNF_MISC_CTL = 0x10a;	
*CMD_SNF_MISC_CTL2 = 0x8400840;	
*CMD_SNF_GF_CTL3 = 0xf000a;	
// Setting NFI part	
*NFI_CON = 0x3;	//NFI Reset
*NFI_CNFG = 0x3005;	//0x3305 if autofmt is enabled, and ECC is enabled.
*NFI_STRADDR = pointer to buffer array;	
*NFI_CMD = 0x80;	
*NFIECC_ENCCNFG = 0x1000_0010;	//ECC related setting. It can be ignored if ECC is disabled.
*NFIECC_ENCCON = 0x0;	
*NFIECC_DECCON = 0x0;	
*NFIECC_ENCCON = 0x1;	
*NFI_FDMXX = fdm_data;	//Set FDM data; this can be removed if autofmt is disabled.
*NFI_CON = 0x4200;	//Trigger burst write and trigger SPI
*NFI_INTR_EN = 0x40;	//Wait for an interrupt
	//Wait for auto program done interrupt (updated later)

- Auto Read Mode (Auto Mode)

Configuration	Memo
*CMD_SNF_MISC_CTL = 0x10a;	
*CMD_SNF_GF_CTL3 = 0xf000a;	
//Setting NFI part	
*NFI_CON = 0x3;	//Reset NFI register status
*NFI_PAGEFMT = 0x2;	//Set pagefmt
*NFI_CNFG = 0x631f;	//Custom mode is a must for SPI_NAND (read_mode is prohibited).
*NFI_CMD = 0;	//Dummy command to trigger the state to custom mode
*NFIECC_DECCNFG = 0x90343110;	//ECC related setting. It can be ignored if ECC is disabled.
*NFIECC_DECCON = 0x0;	
*NFIECC_ENCCON = 0x0;	
*NFIECC_DECCON = 0x1;	
*NFIECC_ENCCNFG = 0x1000_0010;	
*NFIECC_ENCCON = 0x0;	
*NFIECC_DECCON = 0x0;	
*NFIECC_ENCCON = 0x1;	
Handling *NFI_FDMXX data;	//Check FDM data; the data does not exist here if autofmt is disabled.
*NFI_STRADDR = pointer to buffer array;	

Configuration	Memo
*NFI_CON = 0x4100;	//Trigger to start transfer data
*NFI_INTR_EN;	//Wait for an interrupt
	//Wait for auto read done interrupt (updated later)

3.7.3.9 Register Definition

Please refer to "*MT8395 Register Map*" for detailed register descriptions.

3.7.4 Serial NOR Flash Controller (SNFC)

3.7.4.1 Overview

A Serial NOR Flash Controller (SNFC) provides convenient access to high-speed serial NOR flash devices. The SNFC supports:

- Single-bit Serial Peripheral Interface (SPI) serial NOR flash
- High-performance dual-bit and quad-bit SPI serial NOR flash

The SPI clock speed can reach up to 52 MHz for the single-bit SPI, dual-bit SPI and quad-bit SPI. The combination of the SNFC and the serial NOR flash is an important component of system bootup and can also replace DRAM, executing within the NOR flash chip (XIP). SPI can also complete the access of serial NOR flash. The difference between SPI and SNFC is that the SNFC is specifically tailored for NOR flash memory, making it more efficient.

3.7.4.2 Features

- SPI bus compatible serial interface for common serial NOR flash devices.
- Maps out 512-byte page program buffer and supports multi-page program.
- Supports the SPI mode (single-bit) to transfer page program and 1-byte program.
- Supports the 4-byte address mode, 3-byte address mode compatible.
- Supports single-bit read, dual output and dual I/O read, as well as quad output and quad I/O read mode.
- Reads serial NOR flash data through the direct read, PIO read or DMA read mode.
- Supports serial NOR flash device frequency up to 52 MHz.
- Supports serial NOR flash devices MX25U25645G and W25Q256JW.

3.7.4.3 Block Diagram

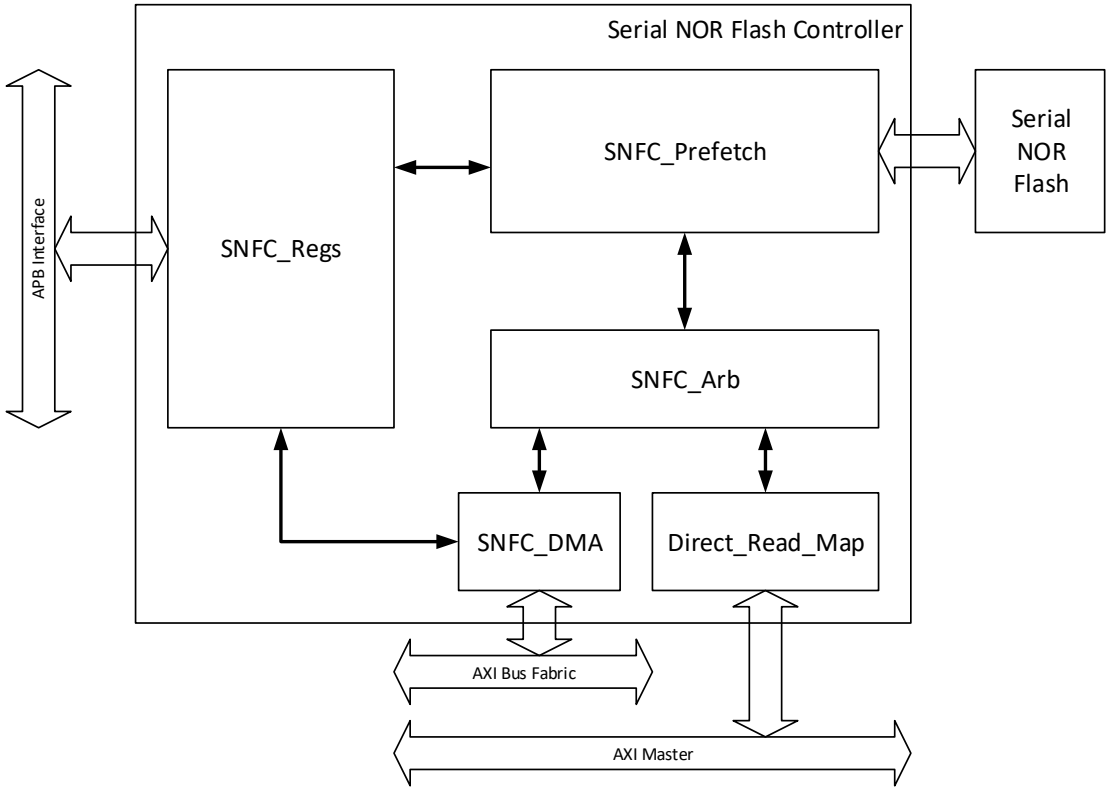


Figure 3-40 Block Diagram of SNFC

Figure 3-40 illustrates the block diagram of the SNFC, which includes the following.

Table 3-28 SNFC Modules

Module	Description
SNFC_Regs	Register control module, which allows the system master to access controller registers through the APB interface.
SNFC_Prefetch	Allocates a buffer with a size of 128 x 32 bits for the reading and programming processes. • During the reading process, data is first transferred to the buffer and then sent to the read master. • In the programming process, the entire page data should be written to the buffer, and the SNFC delivers the data to the serial NOR flash device after the page program is triggered.
SNFC_Arb	Employs an arbitration mechanism when Direct_Read_Map and SNFC_DMA access the serial NOR flash device simultaneously.
SNFC_DMA	A hardware engine that automatically reads data from the serial NOR flash and writes data to the SRAM through the AXI bus. The CPU must configure the source address and start and end destination addresses through the SNFC_Regs module before the DMA, SNFC_DMA, starts.
Direct_Read_Map	Translates the AXI master address and serial NOR flash memory address, and is responsible for returning data to the CPU through the AXI master.

3.7.4.4 Function Description

The SNFC integrates commonly used serial NOR flash operations commands, enabling convenient access to the serial NOR flash. Even without any configuration, it can directly read data from the serial NOR flash. In addition to the conventional reading, the SNFC also supports writing to the serial NOR flash, and configuring serial NOR flash operations.

The SNFC handles all commands, addresses, data sequences and serial interface protocols. It allows reading of serial NOR flash in three ways as stated in Table 3-29.

Table 3-29 Methods to Read Serial NOR Flash

Mode	Description
PIO read mode	The CPU can program the control registers, SNFC_Regs, in a specific sequence and obtain the serial NOR flash data through the APB. This mode is usually used for reading few-byte data.
DMA read mode	The SNFC_DMA copies serial NOR flash data to the SRAM through the AXI bus.
Direct read mode	The CPU can directly read serial flash data through the AXI bus by the address offset.

The SNFC supports the following two ways to write the serial NOR flash.

Table 3-30 Methods to Write Serial NOR Flash

Mode	Description
PIO write mode	The CPU can control the registers in a specific order through the APB, enabling single-byte write operations on the serial NOR flash. This mode is usually used for writing few-byte date.
Buff write mode	The CPU can write data to the SNFC buffer, which holds up to 512 bytes at a time, via the APB to write to a specific register and initiate the transfer. The SNFC can write all data less than 512 bytes at once to the specified address space of the serial NOR flash.

3.7.4.5 Theory of Operations

3.7.4.5.1 Read Serial NOR Flash ID

This operation reads the JEDEC ID of the serial NOR flash, including the 1-byte manufacturer ID and 2-byte device ID. The operation sequence is illustrated in Figure 3-41, and the programming flow is in Table 3-31.

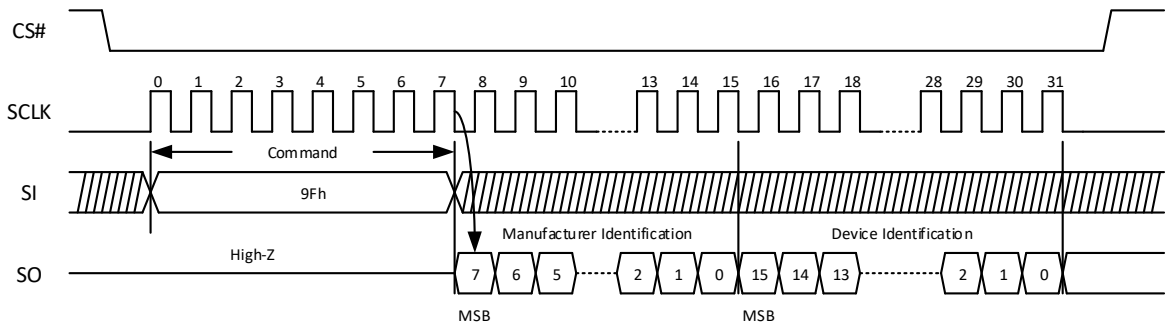


Figure 3-41 Read ID (RDID) Operation Sequence

Table 3-31 Read ID (RDID) Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0xc4	REG_SF_WRPORT	REG_SF_WRPORT[7:0]	W	8'h30	Turn off the controller operation protection (only set once after the controller is reset).
2	SF_Base+0x34	REG_SF_PRGDATA5	REG_SF_PRGDATA5[7:0]	W	8'h9f	Write the operation command of RDID.
3	SF_Base+0x30	REG_SF_PRGDATA4	REG_SF_PRGDATA4[7:0]	W	8'h00	Write dummy data.
4	SF_Base+0x2c	REG_SF_PRGDATA3	REG_SF_PRGDATA3[7:0]	W	8'h00	Write dummy data.
5	SF_Base+0x28	REG_SF_PRGDATA2	REG_SF_PRGDATA2[7:0]	W	8'h00	Write dummy data.
6	SF_Base+0x04	REG_SF_CNT	REG_SF_CNT[5:0]	W	6'h21	Write the process cycle count.
7	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	W	1'b1	Trigger the controller (send the RDID operation sequence to the serial NOR flash).
8	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	R	1'b0	When this bit is 1'b0, the controller processing is done.
9	SF_Base+0x38	REG_SF_SHREG0	REG_SF_SHREG0[7:0]	R		Read the JEDEC ID (device ID - LowByte)
10	SF_Base+0x3c	REG_SF_SHREG1	REG_SF_SHREG1[7:0]	R		Read the JEDEC ID (device ID - HighByte)
11	SF_Base+0x40	REG_SF_SHREG2	REG_SF_SHREG2[7:0]	R		Read the JEDEC ID (manufacturer ID)

3.7.4.5.2 Erase Serial NOR Flash

The SNFC facilitates three erase operations, namely Sector Erase, Block Erase and Chip Erase, for serial NOR flash. These erase operations are executed to clear the data of designated part to “1”. However, before you send the erase command, it is important to execute a WREN (WRITE enable) instruction that sets the WEL (Write Enable Latch).

The Sector Erase command is used for a 4KB sector, while the Block Erase command is used for a 64KB block, and the Chip Erase command for the entire serial NOR flash. The erase-command formats are as follows.

Table 3-32 Erase-command Formats

Erase Method	Format	Operation Sequence	Programming Flow
Section Erase	8-bit operation code (0x20) followed by the 24-bit or 32-bit address.	Figure 3-42	Table 3-33
Block Erase	8-bit operation code (0xd8) followed by the 24-bit or 32-bit address.		
Chip Erase	8-bit operation code (0xc7)	Figure 3-43	Table 3-34

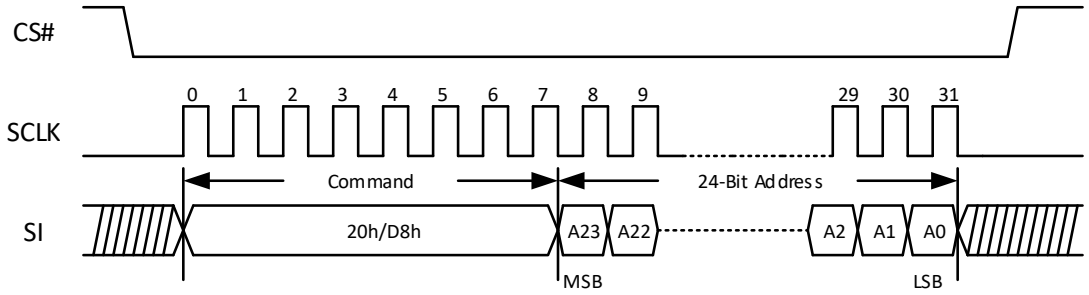


Figure 3-42 Sector Erase and Block Erase Operation Sequence

Table 3-33 Sector Erase and Block Erase Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x34	REG_SF_PRGDATA5	REG_SF_PRGDATA5[7:0]	W	8'h20 or 8'hd8	Write the operation command of Sector Erase (0x20) or Block Erase (0xd8).
2	SF_Base+0x30	REG_SF_PRGDATA4	REG_SF_PRGDATA4[7:0]	W	addr[31:24]	Write erase address bit31:bit24. This register only needs to be set when it is in 4-byte address mode.
3	SF_Base+0x2c	REG_SF_PRGDATA3	REG_SF_PRGDATA3[7:0]	W	addr[23:16]	Write erase address bit23:bit16.
4	SF_Base+0x28	REG_SF_PRGDATA2	REG_SF_PRGDATA2[7:0]	W	addr[15:8]	Write erase address bit15:bit8.
5	SF_Base+0x24	REG_SF_PRGDATA1	REG_SF_PRGDATA1[7:0]	W	addr[7:0]	Write erase address bit7:bit0.
6	SF_Base+0x04	REG_SF_CNT	REG_SF_CNT[5:0]	W	6'h20 or 6'h28	Write the process cycle count: Set 0x20 for 3-byte address mode. Set 0x28 for 4-byte address mode.
7	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	W	1'b1	Trigger the controller to send the erase operation sequence to the serial NOR flash.
8	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	R	1'b0	When this bit is 1'b0, the controller processing is done.
9	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[1]	W	1'b1	(Polling serial NOR flash status) Send read flash status command to serial NOR flash.

Step	Address	Register Name	Local Address	R/W	Value	Description
10	SF_Base+0x08	REG_SF_RDSR	REG_SF_RDSR[0]	R	1'b0	(Polling serial NOR flash status) Check whether the WIP bit of serial NOR flash status is de-asserted. The erase process is completed when this bit is 0.

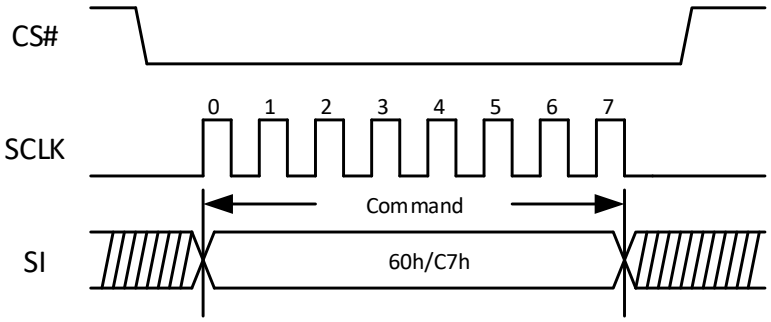


Figure 3-43 Chip Erase Operation Sequence

Table 3-34 Chip Erase Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x34	REG_SF_PRGDATA5	REG_SF_PRGDATA5[7:0]	W	8'hc7	Write the operation command of Chip Erase.
2	SF_Base+0x04	REG_SF_CNT	REG_SF_CNT[5:0]	W	6'h08	Write the process cycle count.
3	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	W	1'b1	Trigger the controller to send the chip erase operation sequence to the serial NOR flash.
4	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	R	1'b0	When this bit is 1'b0, the controller processing is done.
5	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[1]	W	1'b1	(Polling serial NOR flash status.) Send read flash status command to serial NOR flash
6	SF_Base+0x08	REG_SF_RDSR	REG_SF_RDSR[0]	R	1'b0	(Polling serial NOR flash status) Check whether the WIP bit of serial NOR flash status is de-asserted. The erasing process is completed when this bit is 0.

3.7.4.5.3 Program Serial NOR Flash

The programming command is utilized to program the memory to “0”. Prior to initiating the programming process, a WREN command must be executed to set the WEL bit. The SNFC supports two kinds of programming operations: Page Program and PIO Program.

Table 3-35 Methods to Program Seiral NOR Flash

Program Method	Description
Page Program	The maximum data size of the programming operation is limited to the serial NOR flash page size, which is typically 256 bytes for most devices.
PIO Program	The controller register programs one byte at a time.

Figure 3-44 shows the programming operation sequence.

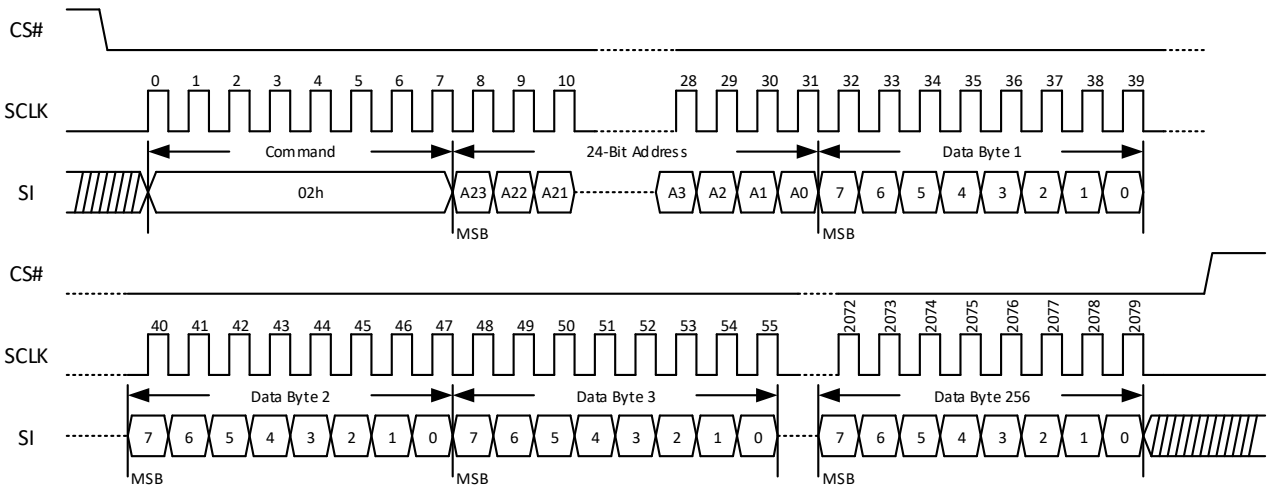


Figure 3-44 Programming Operation Sequence

Furthermore, there are three cases of the two programming operations mentioned above.

1. Entire page programming via the SNFC_Prefetch buffer (3.7.4.6)
2. Entire multi-page programming via the SNFC_Prefetch buffer (3.7.4.7)
3. Writing data to the serial flash 1 byte at a time (PIO write mode) (3.7.4.8)

3.7.4.6 Entire Page Programming via SNFC_Prefetch Buffer

Table 3-36 Page Program (PP) Operation Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x64	REG_SF_CFG2	REG_SF_CFG2[0]	W	1'b1	Enable the SNFC_Prefetch buffer for writing.
2	SF_Base+0x10	REG_SF_RADR0	REG_SF_RADR0[7:0]	W	addr[7:0]	Write address bit7:bit0
3	SF_Base+0x14	REG_SF_RADR1	REG_SF_RADR1[7:0]	W	addr[15:8]	Write address bit15:bit8
4	SF_Base+0x18	REG_SF_RADR2	REG_SF_RADR2[7:0]	W	addr[23:16]	Write address bit23:bit16
5	SF_Base+0xc8	REG_SF_RADR3	REG_SF_RADR3[7:0]	W	addr[31:24]	Write address bit31:bit24. This register only needs to be set when it is in 4-byte address mode.

Step	Address	Register Name	Local Address	R/W	Value	Description
6	SF_Base+0x98	REG_SF_PP_DW_DATA	REG_SF_PP_DW_DATA[31:0]	W	Data[31:0]	Fill the SNFC_Prefetch buffer by writing program data to this register. Loop N times (N indicates that data length imported must be an integer multiple of 32 bytes, and the maximum data length is 256 bytes).
7	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[4]	W	1'b1	Trigger page program controller process
8	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[4]	R	1'b0	When this bit is 1'b0, the controller processing is done.
9	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[1]	W	1'b1	Polling serial NOR flash status. Send read flash status command to serial NOR flash
10	SF_Base+0x08	REG_SF_RDSR	REG_SF_RDSR[0]	R	1'b0	Polling serial NOR flash status. Check whether the WIP bit of serial NOR flash status is de-asserted or not. The page program processing is completed when this bit is 0.

3.7.4.7 Entire Multi-Page Programming via SNFC_Prefetch Buffer

Table 3-37 Multi-Page Program (MPP) Operation Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x64	REG_SF_CFG2	REG_SF_CFG2[0]	W	1'b1	Enable the SNFC_Prefetch buffer for writing.
2	SF_Base+0x10	REG_SF_RADR0	REG_SF_RADR0[7:0]	W	addr[7:0]	Write program address bit7:bit0
3	SF_Base+0x14	REG_SF_RADR1	REG_SF_RADR1[7:0]	W	addr[15:8]	Write program address bit15:bit8
4	SF_Base+0x18	REG_SF_RADR2	REG_SF_RADR2[7:0]	W	addr[23:16]	Write program address bit23:bit16
5	SF_Base+0xc8	REG_SF_RADR3	REG_SF_RADR3[7:0]	W	addr[31:24]	Write program address bit31:bit24. This register only needs to be set when it is in 4-byte address mode.

Step	Address	Register Name	Local Address	R/W	Value	Description
6	SF_Base+0x72c	REG_SF_PAGECNT	REG_SF_PAGECNT[7:0]	W	page_cnt[7:0]	Set the total number of pages needed to be programed in SNFC_Prefetch buffer
7	SF_Base+0x730	REG_SF_PAGESIZE	REG_SF_PAGESIZE[7:0]	W	page_size[7:0]	Set the page size of serial NOR flash
8	SF_Base+0x734	REG_MPP_EN	REG_MPP_EN[0]	W	1'b1	Enable MPP function
9	SF_Base+0x98	REG_SF_PP_DW_DATA	REG_SF_PP_DW_DATA [31:0]	W	data[31:0]	Fill the SNFC_Prefetch buffer by writing program data to this register. Loop N times. (N indicates that the data length imported must be an integer multiple of 32 bytes. The maximum data length is 512 bytes).
10	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[7]	W	1'b1	Address increases automatically when programming; keep this bit high during the entire programming process
11	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[4]	W	1'b1	Trigger multi-page program controller process
12	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[4]	R	1'b0	When this bit is 1'b0, the controller processing is done.
13	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[1]	W	1'b1	Polling serial NOR flash status. Send read flash status command to serial NOR flash
14	SF_Base+0x08	REG_SF_RDSR	REG_SF_RDSR[0]	R	1'b0	Polling serial NOR flash status. Confirm whether the WIP bit of serial NOR flash status is de-asserted or not. The multi-page program processing is completed when this bit is 0.

3.7.4.8 Write Data to Serial Flash of 1 Byte (PIO Write Mode)

Table 3-38 PIO Write Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x1c	REG_SF_WDATA	REG_SF_WDATA[7:0]	W	data[7:0]	One byte data needs to be programed.
2	SF_Base+0x10	REG_SF_RADR0	REG_SF_RADR0[7:0]	W	addr[7:0]	Write program address bit7:bit0
3	SF_Base+0x14	REG_SF_RADR1	REG_SF_RADR1[7:0]	W	addr[15:8]	Write program address bit15:bit8
4	SF_Base+0x18	REG_SF_RADR2	REG_SF_RADR2[7:0]	W	addr[23:16]	Write program address bit23:bit16
5	SF_Base+0xc8	REG_SF_RADR3	REG_SF_RADR3[7:0]	W	addr[31:24]	Write program address bit31:bit24. This register only needs to be programed when in 4-byte address mode.
6	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[4]	W	1'b1	Trigger the PIO write controller.
7	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[4]	R	1'b0	When this bit is 1'b0, the controller processing is done.
8	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[1]	W	1'b1	Polling serial NOR flash status. Send read flash status command to serial NOR flash
9	SF_Base+0x08	REG_SF_RDSR	REG_SF_RDSR[0]	R	1'b0	Polling serial NOR flash status. Confirm whether the WIP bit of serial NOR flash status is de-asserted or not. The PIO write process is completed when this bit is 0.

3.7.4.8.1 Read Flash

The SNFC provides three modes for reading data from the flash device:

- PIO read mode (3.7.4.9)
- DMA read mode (3.7.4.10)
- Direct read mode (3.7.4.11)

The standard SPI link format is combined with command, address and data bytes. The read operation format and sequence of the single-bit SPI mode is depicted in Figure 3-45. Additionally, the controller supports dual output, dual I/O, quad output, and quad I/O read modes. For the corresponding register configuration, refer to *REG_SF_DUAL (SF_Base+0xcc)* in Section 3.7.4.12.5.

For a 4-byte address, set LARGE_ADDR_EN (SF_Base+0xcc [4]) to enable the address cycle to increase from 24 bits to 32 bits.

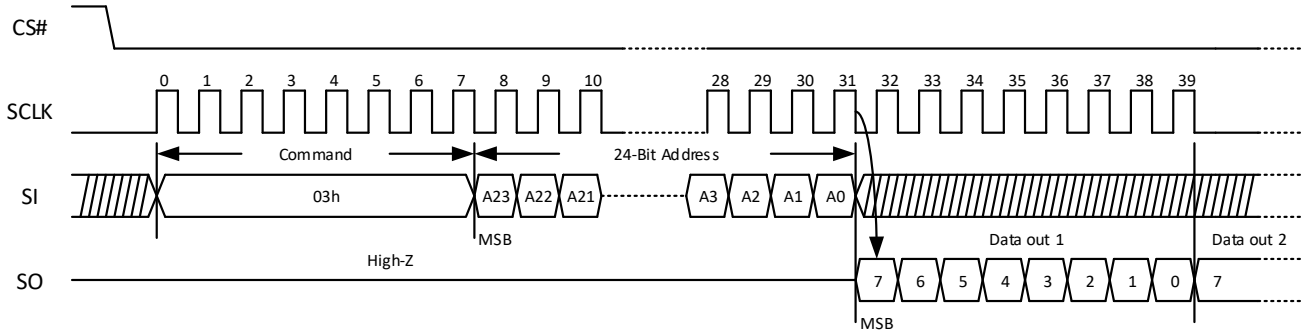


Figure 3-45 Read Operation Sequence

3.7.4.9 PIO Read Mode

Table 3-39 PIO Read Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x10	REG_SF_RADR0	REG_SF_RADR0[7:0]	W	addr[7:0]	Write read start address bit7:bit0
2	SF_Base+0x14	REG_SF_RADR1	REG_SF_RADR1[7:0]	W	addr[15:8]	Write read start address bit15:bit8
3	SF_Base+0x18	REG_SF_RADR2	REG_SF_RADR2[7:0]	W	addr[23:16]	Write read start address bit23:bit16
4	SF_Base+0xc8	REG_SF_RADR3	REG_SF_RADR3[7:0]	W	addr[31:24]	Write read start address bit31:bit24. This register only needs to be programed when in 4-byte address mode.
5	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[7]	W	1'b1	Address increases automatically in the PIO mode.
6	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[0]	W	1'b1	Trigger the controller in the PIO mode.
7	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[0]	R	1'b0	When this bit is 1'b0, the controller processing is done.
8	SF_Base+0x0c	REG_SF_RDATA	REG_SF_RDATA[7:0]	R	data[7:0]	Get the serial NOR flash data (one byte).

3.7.4.10 DMA Read Mode

The hardware engine supports copying data from the serial NOR flash to the SRAM via SNFC_DMA. Prior to triggering SNFC_DMA, the source address, destination start address and end address must be appropriately configured. Once initiated, the hardware engine automatically copies data from the serial NOR flash device to the designated destination address.

Table 3-40 DMA Read Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x71C	REG_FDMA_FADR	REG_FDMA_FADR[31:0]	W	src_addr[31:0]	Set the DMA source address.

Step	Address	Register Name	Local Address	R/W	Value	Description
2	SF_Base+0x720	REG_FDMA_DADR	REG_FDMA_DADR[31:4]	W	des_addr[31:4]	Set the DMA destination start address.
3	SF_Base+0x724	REG_FDMA_END_DADR	REG_FDMA_END_DADR[31:4]	W	des_end_addr[31:4]	Set the DMA destination end address.
4	SF_Base+0x718	REG_FDMA_CTL	REG_FDMA_CTL[0]	W	1'b1	Trigger the DMA module process.
5	SF_Base+0x718	REG_FDMA_CTL	REG_FDMA_CTL[0]	R	1'b0	Poll the trigger bit as 0 to indicate that the DMA processing is completed.

3.7.4.11 Direct Read Mode

The CPU can directly read serial NOR flash data through AXI bus. For instance, the CPU can directly issue the data address and length, and the SNFC returns the corresponding data. The application supports the following operations:

- The system executes the memory copy function via the CPU data instruction.
- The system DMA (e.g., Crypto DMA) requests serial NOR flash data.

3.7.4.12 Quad-Bit Read Mode

The SNFC supports the quad-bit SPI read mode to enhance read performance, which includes the quad output mode and quad I/O read mode. The format is similar to the single-bit SPI link format, with the addition of a dummy cycle.
This format comprises of:

- Command
- Address (quad output single-bit; quad I/O quad-bit)
- Dummy cycle (quad output for at least eight dummy cycles; quad I/O for at least six dummy cycles)
- Data

Note that the definition of the QE bit mentioned in this section may differ based on the flash vendor. Refer to Serial NOR Flash Datasheet for more information, such as MX25L25635F series, W25Q256JW series.

The differences in the address and dummy cycle format between the quad output and quad I/O are illustrated in Figure 3-46 and Figure 3-47, respectively.

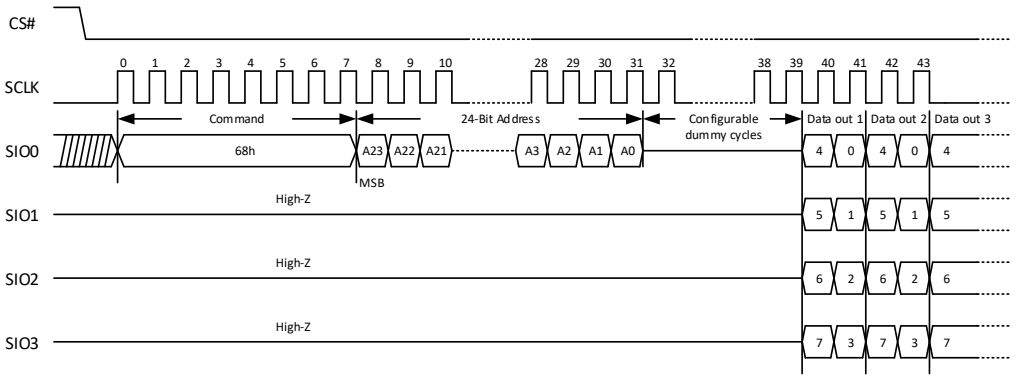


Figure 3-46 Quad Output Read Mode Sequence (Address Sent in the Single-bit Mode)

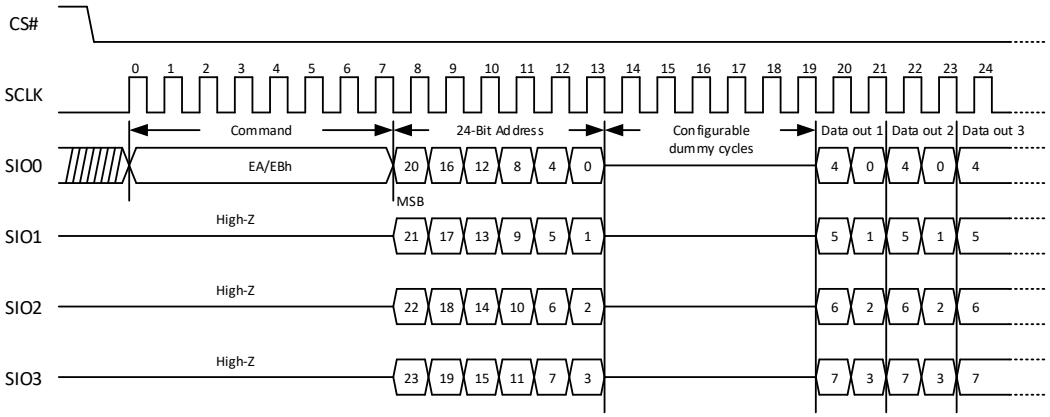


Figure 3-47 Quad I/O Read Mode Sequence (Address Sent in the 4-bit Mode)

3.7.4.12.1.1.1 Enter Quad-Bit Read Mode

To enter the quad-bit read mode, refer to the programming flow as below:

1. Enable the Quad Enable (QE) bit of the serial NOR flash.
2. Set REG_SF_DUAL (SF_Base+0xcc) [3] and [2] = 1b'1 to enable the SNFC quad-bit mode.

3.7.4.12.1.1.2 Exit Quad-Bit Read Mode

To exit the quad-bit read mode, refer to the programming flow as below:

1. Set REG_SF_DUAL (SF_Base+0xcc)[3] and [2] = 1b'0 to disable the SNFC quad-bit mode.
2. Disable the QE bit of the serial NOR flash

3.7.4.12.1.2 Quad-Byte Address Mode

For most serial NOR flash devices, the default address is a 3-byte address. The following sections introduce how to enter and exit space larger than 16MB.

3.7.4.12.1.2.1 Enter Quad-Byte Address Mode

Table 3-41 Enter Quad-byte Address Mode Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0x34	REG_SF_PRGDATA5	REG_SF_PRGDATA5[7:0]	W	8'hb7	Write the operation command of entering quad-byte address mode.
2	SF_Base+0x04	REG_SF_CNT	REG_SF_CNT[5:0]	W	6'h08	Write the process cycle count.
3	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	W	1'b1	Trigger the controller process (send OP Code to serial NOR flash).
4	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	R	1'b0	When this bit is 1'b0, the controller processing is done.
5	SF_Base+0xcc	REG_SF_DUAL	REG_SF_DUAL[4]	W	1'b1	Enable controller quad-byte address mode

3.7.4.12.1.2.2 Exit Quad-Byte Address Mode

Table 3-42 Exit Quad-byte Address Mode Operation Programming Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
1	SF_Base+0xcc	REG_SF_DUAL	REG_SF_DUAL[4]	W	1'b0	Disable the controller quad-byte address mode.
2	SF_Base+0x34	REG_SF_PRGDATA5	REG_SF_PRGDATA5[7:0]	W	8'he9	Write the operation command of exiting quad-byte address mode.
3	SF_Base+0x04	REG_SF_CNT	REG_SF_CNT[5:0]	W	6'h08	Write the process cycle count.
4	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	W	1'b1	Trigger the controller process (send OP Code to serial NOR flash).
5	SF_Base+0x00	REG_SF_CMD	REG_SF_CMD[2]	R	1'b0	When this bit is 1'b0, the controller processing is done.

3.7.4.12.1.3 Recovery Sequence

To avoid synchronization problems between the SNFC and serial NOR flash device following an entire chip reset, refer to the planned recovery sequence as follows:

1. Release power down (OP code 0xab, SPI) to wake up the serial NOR flash device and avoid entering the power down mode.
2. Perform a software reset of the device (OP code 0x66/0x99, SPI) to reset the serial NOR flash device.
3. Initialize the serial NOR flash device by reading the JEDEC ID and detecting the serial NOR flash type.
4. Disable the serial NOR flash protection and send WRSR (OP code 0x01) to clear the protection bit.
5. Determine whether to enter the quad-byte address mode or not (OP Code 0xb7), depending on the serial NOR flash size (>16 MB).
6. Set the serial NOR flash operation mode (SPI/SPI-D/SPI-Q) and enable or disable the QE bit using the WRSR (OP code 0x01) operation.

7. Set the SNFC register (including the control bit and OP code setting) for SPI/SPI-D/SPI-Q modes.

3.7.4.12.2 Programming Guide

The SF_Base is a specific physical address at 0x11018000.

Table 3-43 Overall SNFC Programming Guide

Number	Purpose/Mode	Programming Guide
1	Read serial nor flash ID	Table 3-31
2	Erase serial NOR Flash-sector/block	Table 3-33
3	Erase serial NOR Flash-chip	Table 3-34
4	Program serial NOR flash-PP mode	Table 3-36
5	Program serial NOR flash-MPP mode	Table 3-37
6	Program serial NOR flash-PIO write mode	Table 3-38
7	Read Flash-PIO read mode	Table 3-39
8	Read Flash-DMA read mode	Table 3-40
9	Read Flash-Direct read mode	N/A
10	Enter quad-bit read mode	3.7.4.12.1.1.1
11	Exit quad-bit read mode	3.7.4.12.1.1.2
12	Enter quad-byte address mode	Table 3-41
13	Exit quad-byte address mode	Table 3-42
14	Recovery sequence	3.7.4.12.1.3

3.7.4.12.3 SNOR Signal Descriptions

Table 3-44 presents SNOR signal descriptions.

Table 3-44 SNOR Signal Descriptions

Signal Name	Type	Description	Ball Location
SPINOR_CK	DO	SNOR clock	AP22
SPINOR_CS	DO	SNOR chip select	AN23
SPINOR_IO0	DIO	SNOR I/O data 0 (MOSI)	AN22
SPINOR_IO1	DIO	SNOR I/O data 1 (MISO)	AP23
SPINOR_IO2	DIO	SNOR I/O data 2 (WP)	AL7
SPINOR_IO3	DIO	SNOR I/O data 3 (hold)	AL8

3.7.4.12.4 SNOR Timing Characteristics

Table 3-45 and Figure 3-48 present the SNOR timing characteristics.

Table 3-45 SNOR Timing Characteristics

No	Parameter(1)	Description	Min	Max	Unit
SNOR01	F_{ck}	SNOR clock (SF_SCLK) frequency	-	25	MHz
SNOR02	D	Duty Cycle, SF_SCLK	45	55	%

No	Parameter[1]	Description	Min	Max	Unit
SNOR03	t_{CLQx}	Input setup time	4.635	-	ns
SNOR04	t_{CLQV}	Input hold time	-4.82	-	ns
SNOR05	t_d	Output delay	-	2.151	ns
SNOR06	t_h	Output hold time	-0.37	-	ns
SNOR07	$t_{R\_CS/SCLK}$	CS low to SF_SCLK rising edge (read)	1.5/SCLK	9.5/SCLK	ns
SNOR08	$t_{R\_SCLK/CS}$	SF_SCLK falling edge to CS high (read)	3/SCLK	10/SCLK	ns

(1) The specification is based on the assumed load capacity value of 30 pF.

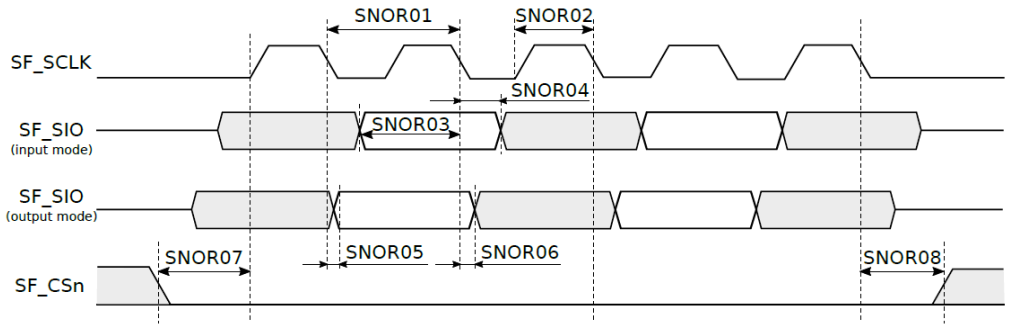


Figure 3-48 SNOR Timing Diagram

3.7.4.12.5 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.7.4.12.6 Reference

Serial NOR Flash Datasheet, such as MX25L25635F series, W25Q256JW series, etc.

3.8 Display

3.8.1 Display Controller

3.8.1.1 Overview

The Display Controller, also referred to as VPPSYS, is the abbreviation of "Video Data Processing Pipe Subsystem." It is used to support camera video post processing, video decoder post processing, video preprocessing for video encoder or APU and pixel data processing for Display. The data processing includes resize, sharpen, composition, HDR, warpage, color, gamma, etc. VPPSYS also contains multiple video IO interfaces, including HDMI_RX, HDMI_TX, DP_TX, eDP_TX, DSI, and Digital video in and out. Data processing pipes can be directly linked with these video IO interfaces for low latency experience.

VPPSYS consists of three hardware parts, SVPP, DVPP and WPE. S stands for time-share. D stands for display. WPE is for image warpage. The following sections show the features of each hardware part and the operation modes.

3.8.1.2 Features

SVPP-0 and SVPP-1:

- Read DRAM agent (Supports AFBC)
- Film grain noise for supporting AV1 video
- HDR decode (HDR10, HDR10+, HLG)
- Local contrast enhancement
- Image resizer and sharpness
- Preference color enhancement
- Multi-layer image compositors
- Image aspect ratio converter
- HDR10+ metadata statistic
- Dual image horizontal stitching (in-line with WPE)
- Write DRAM agent with image rotation { 90, 180, 270, left right flip } (support AFBC)
- The max pixel throughput of each SVPP pipe is 420 Mpixels/sec

SVPP-2 and SVPP-3:

- With similar features as SVPP-0,1. Without image compositor and HDR10+ metadata statistic block.
- The max pixel throughput of each SVPP pipe is 420 Mpixels/sec

WPE:

- Backward grid-map based image warpage
- Max map size: 640x640
- Two data processing cores; each can process 300 Mpixels/sec

DVPP-0:

- 5-layer (1 for video from SVPP, 4 for graphics from DRAM) RGB image compositor
- Graphic layer support AFBC
- Preference color adjustment
- Global contrast adjustment
- Display CSC, Gamma, Dither
- MIPI DSC display data compression
- Two display data processing pipes
- Max image width for each display data processor pipe is 1920 pixels
- Max pixel throughput of each display data processor pipe is 450 Mpixels/sec

DVPP-1:

- Two read DRAM agents (support AFBC) that support Dolby video HDR (Dolby HDR and HDR10+)
- Two read DRAM agents (support AFBC) that support Dolby graphic SDR to HDR
- 4-layer image compositor compliant with Dolby HDR requirements
- The max supported video definition of DVPP-1 is 4K60 (3840x2160x60fps)

Display input and output interfaces:

- HDMI_RX: HDMI2.0 Max 4K60. Max 12-bit color. Support VRR and HDR metadata
- Digital video input: Supports BT656 and CCIR601. 16-bit data bus. 8-bit color depth. Max video definition: FHD60
- DSI_TX: MIPI DSI C(D)PHY combo. Dual link. Maximum lane speed: 1.2 Gbps

- eDP_TX: 4 lanes. Max 5.4 Gbps per lane. 10-bit color. Max 4K60
- DP_TX: DP1.4. 2 lanes. Max 8.1 Gbps per lane. 10-bit color. Max 4K60
- HDMI_TX: HDMI2.0 Max 4K60. Max 12-bit color. Supports VRR and HDR metadata

3.8.1.3 Block Diagram

Figure 3-49 shows the application position of VPPSYS inside the SoC. Figure 3-50 shows the possible application connection of SVPP-x and WPE. Figure 3-51 shows the possible application connection of DVPP-x and Display output interfaces. The best pixel throughput of each display IO interface is also shown in the diagram. Note that only the orange blocks are included in the scope of VPPSYS documentation.

Figure 3-52 shows an example application connection. This example application combines the video conference interfaced through on-device display and HDR video playback output through HDMI connection.

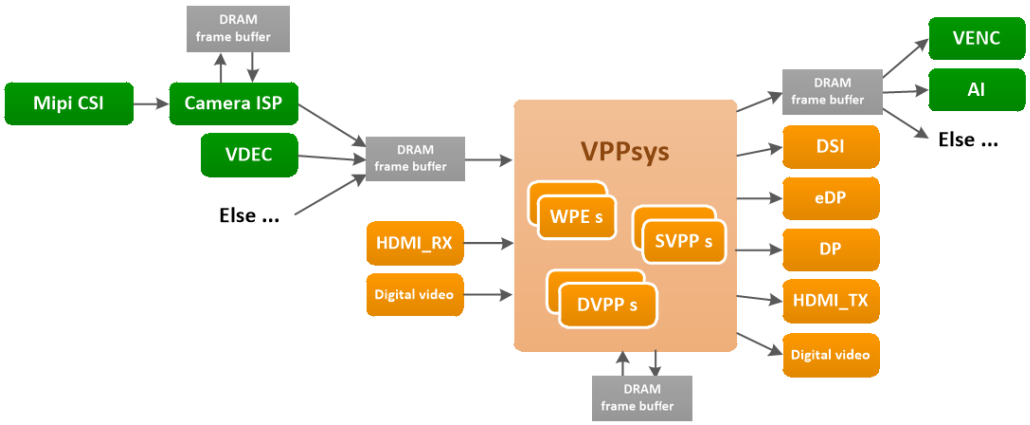


Figure 3-49 Application Position of VPPSYS inside SoC

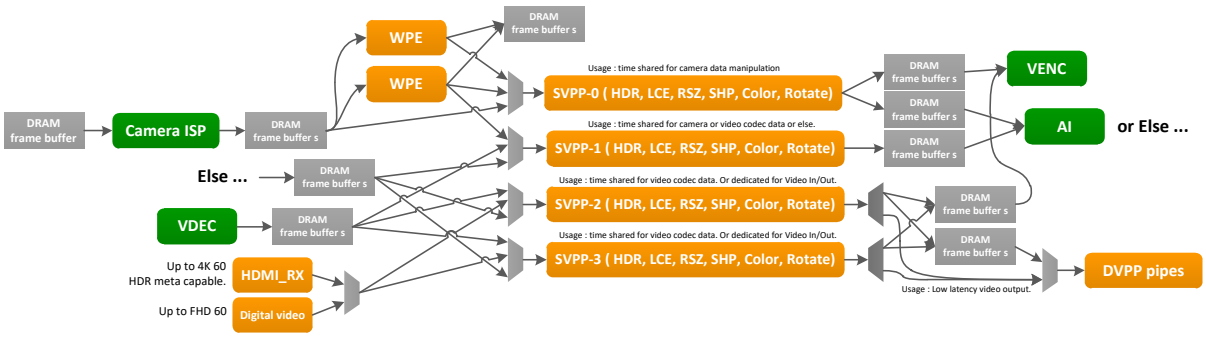


Figure 3-50 Possible Application Connection of SVPP-x and WPE

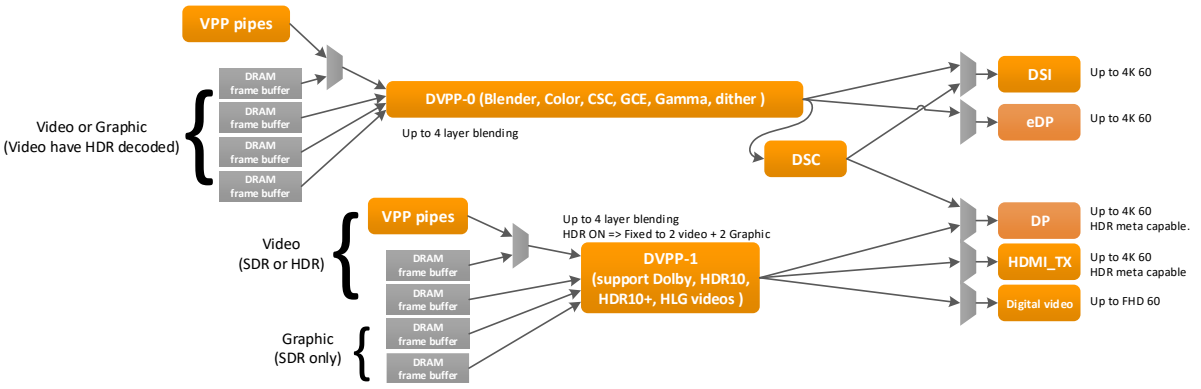


Figure 3-51 Possible Application Connection of DVPP-x and Display Output Interfaces

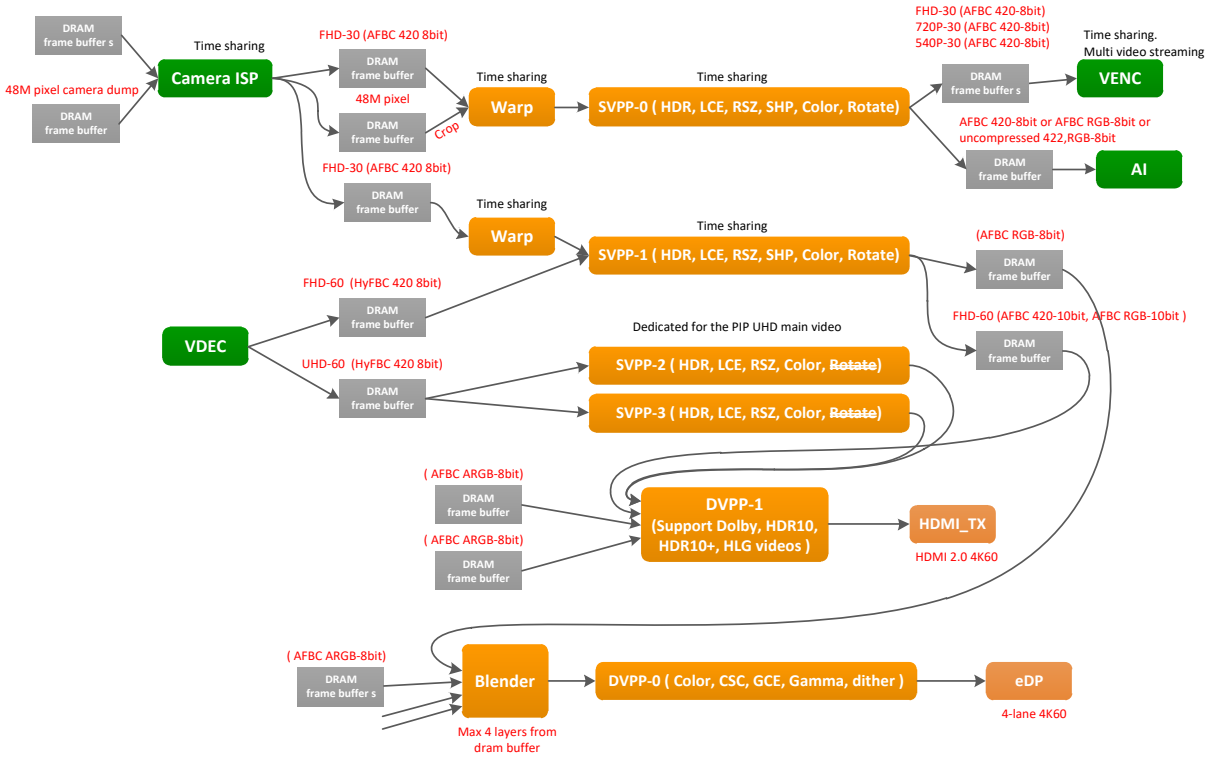


Figure 3-52 Example Application Connection

3.8.1.4 Function Description

SVPP: It is a multipurpose image data processor. The data input could be from DRAM, WPE or Video-input-interfaces (HDMI_RX, Digital-video-input). The data output target could be DRAM or DVPP. Main features of SVPP are image-resizer, image-sharper, HDR decode, composition, image rotation.

DVPP: There are two kinds of DVPP in VPPSYS. DVPP-0 is for SDR display on integrated display panel (DSI, eDP) or external display (DP, HDMI). DVPP-1 is for HDR (Dolby HDR, HDR10+) external display (DP, HDMI). There are two display data processors inside DVPP-0. They can be combined for a single target display output or separated for two independent display outputs. DVPP-0 supports conventional display data processor features like color, gamma, dither, while DVPP-1 is dedicated to standard HDR (Dolby HDR, HDR10+) display. Color or gamma tuning has to be merged with HDR data processing.

WPE: It is a free-form image warpage accelerator. The input is the source image and warpage map. The map is an array of grids in rectangle arrangement. Each grid contains a backward vector pointing out the coordinate in the source image that needs to be warped to the new position in the destination image. The output of WPE can be SVPP or DRAM buffer.

3.8.1.5 Theory of Operations

There are three operation modes in VPPSYS, SW mode, Video mode, and DSI-CMD mode. Each data processor hardware (SVPP-x, WPE, DVPP-0, DVPP-1) can be operated in either one of these three operation modes.

SW mode: As the name implies, the software initiates the hardware operation (one image frame), waits for the HW job done signal (INTR) and then updates the hardware setting of the next job and kicks off again.

Video mode: In this operation mode, the hardware operation is tied to the video timing of Video IO interfaces, including HDMI_RX, HDMI_TX, DSI (video mode), DP_TX and eDP_TX. The video timing generator inside these video IO blocks will create Vsync and VDE signals for hardware data processor pipe. By these signals, the hardware automatically and repeatedly knows when to start a new image frame. To stop and reset the hardware, the software receives the timing signal, by INTR, to know the period that allows change of hardware settings or information read back.

DSI-CMD mode: This mode is available when using DSI command mode display panel. A TE (Tearing Effect) signal from display panel is received by the SoC, which then informs the software by INTR signal. The software, depending on the necessary system operation, updates the hardware setting with the new image frame, which is going to replace the image data inside the frame buffer in display panel.

3.8.2 Display Serial Interface (DSI)

3.8.2.1 Overview

The DSI is based on MIPI Alliance Specification, supporting high-speed serial data transfer between host processor and peripheral devices such as display modules. The device includes two DSI controllers, DSI0 and DSI1. DSI should work with the MIPI_TX_Config module to obtain its engine clock from Analog PHY Macro. And it should work with DMA engines in the previous stage of display path to read out frame pixels from memory, performs frames packing and lane distribution, and then sends the data to a dedicated MIPI D-PHY/C-PHY TX core for serializing.

3.8.2.2 Features

Each DSI controller provides the following key features:

- Compliance with MIPI DSI Specification v01-02-00
- Supports video and command mode data transfers
- Pixel formats supported: RGB888/RGB101010/compressed pixel stream
- 128-entry command queue for command transmission
- 3 types of video modes: Sync-event, sync-pulse, and burst modes
- Limited high-speed residual packet transmission during video mode blanking period
- Ultra-low power mode control
- Peripheral and external Tearing Effect (TE) signals detection

- MIPI D-PHY interface, with the following features:
  - 1 clock lane and up to 4 data lanes, 2 ports
  - Throughput up to 1.2 Gbps per data lane
  - Bi-directional data transmission in Low-Power mode for data lane 0
  - Uni-directional data transmission in High-Speed mode for data lanes 0 through 3
  - Non-continuous high-speed transmission for clock and data lanes
  - Lane swapping
  - Compliance with MIPI D-PHY Specification v1.1
- MIPI C-PHY interface, with the following features:
  - Single 3-trio, 2 ports
  - Throughput up to 1.1 Gsps per trio
  - Trio swapping
  - Compliance with MIPI C-PHY Specification v1.0
- Each D-PHY/C-PHY TX core provides the following main features:
  - Sigma-Delta Modulation (SDM) PLL configuration
  - Spread Spectrum Clocking (SSC) control

3.8.2.3 Block Diagram

Figure 3-53 shows the block diagram of DSI modules. The whole DSI block is constructed by the DSI controller, PHY digital controller and analog PHY macro. The DSI controller implements MIPI DSI application and protocol layers, which mainly focus on packet format, checksum, ECC (Error Correction Code) generation, etc. The MIPI TX configuration module, or called MIPI_TX_Config here, is used to control MIPI TX related registers for lane swap function and analog PHY macro. Lane swap function in MIPI_TX_Config is used to select the order of data and clock lanes. Analog PHY macro converts digital signals to analog signals.

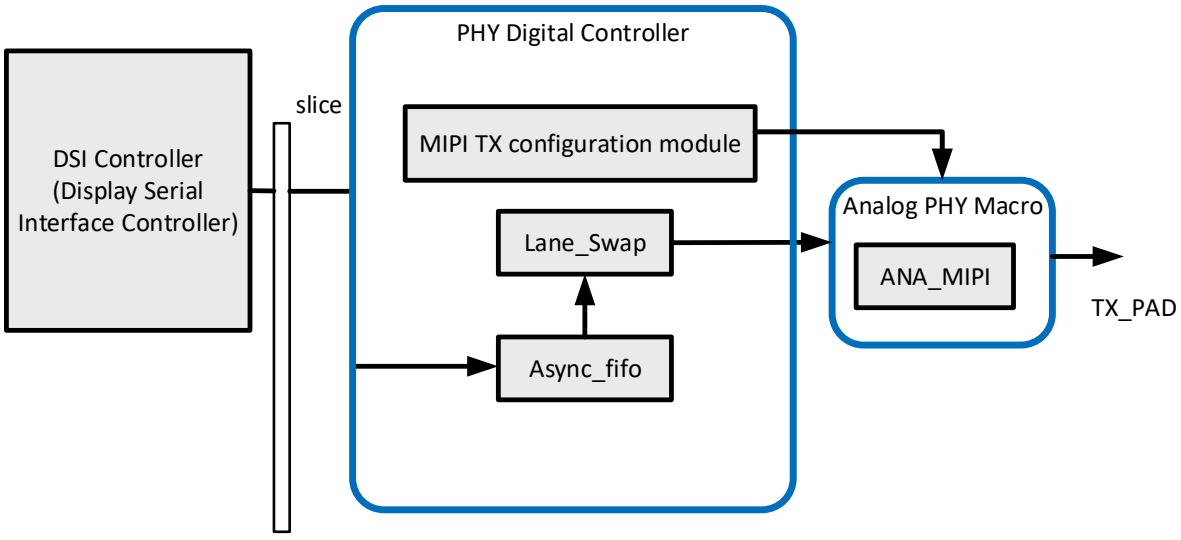


Figure 3-53 DSI Module Block Diagram

3.8.2.4 Function Description

3.8.2.4.1 Size Control

The horizontal size of input data received by DSI is defined by DSI register DSI_WIDTH[12:0]. The unit of DSI_WIDTH[12:0] is pixel. The vertical size of input data received by DSI is defined by DSI register DSI_HEIGHT[12:0]. The unit of DSI_HEIGHT[12:0] is line. The payload length transferred by DSI is defined by DSI register PS_WC[14:0]. The unit of PS_WC[14:0] is byte. The line number transferred by DSI is defined by DSI register VACT_NL[14:0]. The unit of VACT_NL[14:0] is line.

The PS_WC[14:0] is calculated according to DSI_WIDTH[12:0] and PS_SEL[3:0]. The formula of PS_WC[14:0] are
DSI_WIDTH[12:0]*(24/8) for RGB888,
DSI_WIDTH[12:0]*(30/8) for RGB101010,
DSI_WIDTH]12:0]*(24/8) – n for compress (n=0,1,2, payload length sometimes may not be a multiple of three after DSC compression)

3.8.2.4.2 Data Format Control

DSI supports RGB888, RGB101010 and compress mode. The data format is defined by DSI register PS_SEL[3:0](3: RGB888, 4: RGB101010, 5: compress). The compress mode is used in DSC compress.

3.8.2.4.3 Mode Control

DSI supports video and command mode, which is defined by DSI register MODE_CON[1:0].
2'b00: command mode
2'b01: video mode: sync-pulse mode
2'b10: video mode: sync-event mode
2'b11: video mode: burst mode

DSI also supports D-PHY or C-PHY transmission, which is defined by DSI register CPHY_EN.
0: DPHY mode
1: CPHY mode

3.8.2.4.4 Lane Number Control

DSI supports one to four lanes for D-PHY transmission. DSI supports one to three trios for C-PHY transmission, which is defined by DSI register LANE_NUM[3:0].
4'b0001: 1 lane or 1 trio
4'b0011: 2 lanes or 2 trios
4'b0111: 3 lanes or 3 trios
4'b1111: 4 lanes

3.8.2.5 Theory of Operations

MediaTek DSI peripherals support either of two basic modes of operation: command mode and video mode. Which mode is used depends on the architecture and capabilities of the peripheral. The mode definitions reflect the primary intended use of DSI for display interconnection, but they are not intended to restrict DSI from operating in other applications.

3.8.2.5.1 Command Mode

Command mode refers to an operation in which transactions primarily take the form of sending commands and data to a peripheral, such as a display module, which incorporates a display controller. In command mode, long and short packets can be sent to the peripheral, and data can be read/written from/to frame buffer of the peripheral. Command mode operation requires a bidirectional interface.

3.8.2.5.2 Video Mode

Video mode refers to an operation in which transfers from the host processor to the peripheral take the form of a real-time pixel stream. Video information should only be transmitted using HS mode. In video mode, sync-event, sync-pulse and burst modes are supported. These terms are used throughout the following sections:

- Sync-Pulse – Enables the peripheral to accurately reconstruct original video timing, including sync pulse widths.
- Sync-Event – Similar to the one above, but accurate reconstruction of sync pulse widths is not required.
- Burst Mode – RGB pixel packets are time-compressed, leaving more time during a scan line for LP mode (saving power) or for multiplexing other transmissions onto the DSI link.

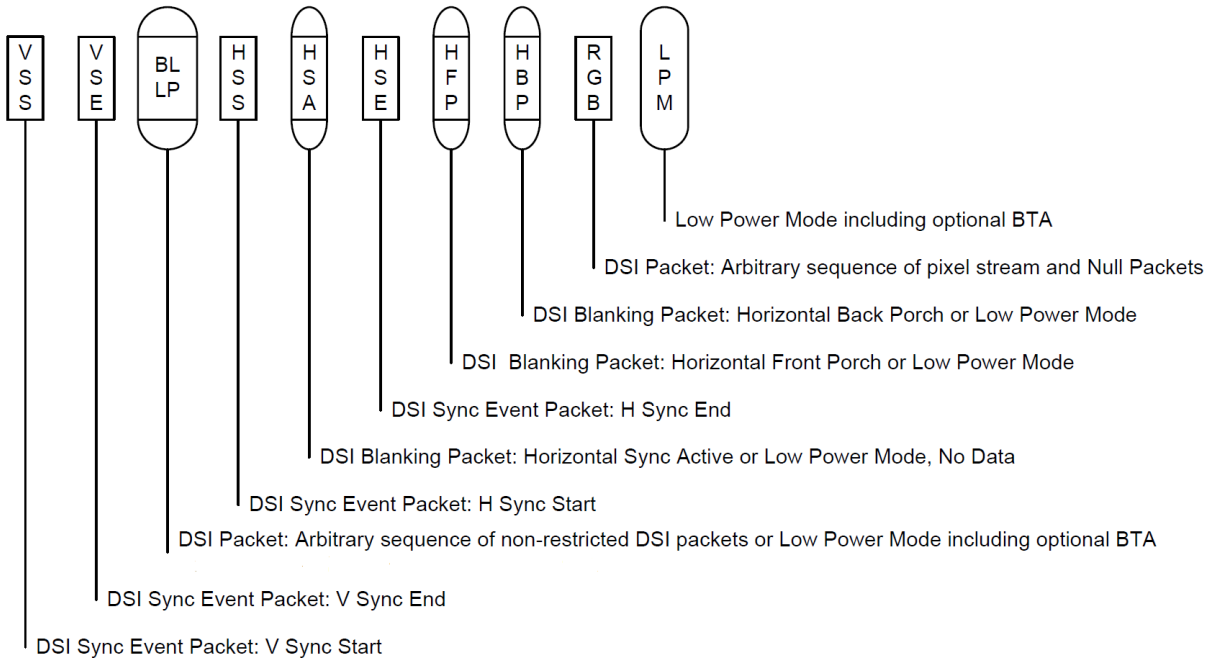


Figure 3-54 Video Mode Interface Timing Legend

Detailed timing and packet sequence are shown in Figure 3-55, Figure 3-56 and Figure 3-57.
Transmission packet components used in Figure 3-55, Figure 3-56 and Figure 3-57 are defined in Figure 3-54.

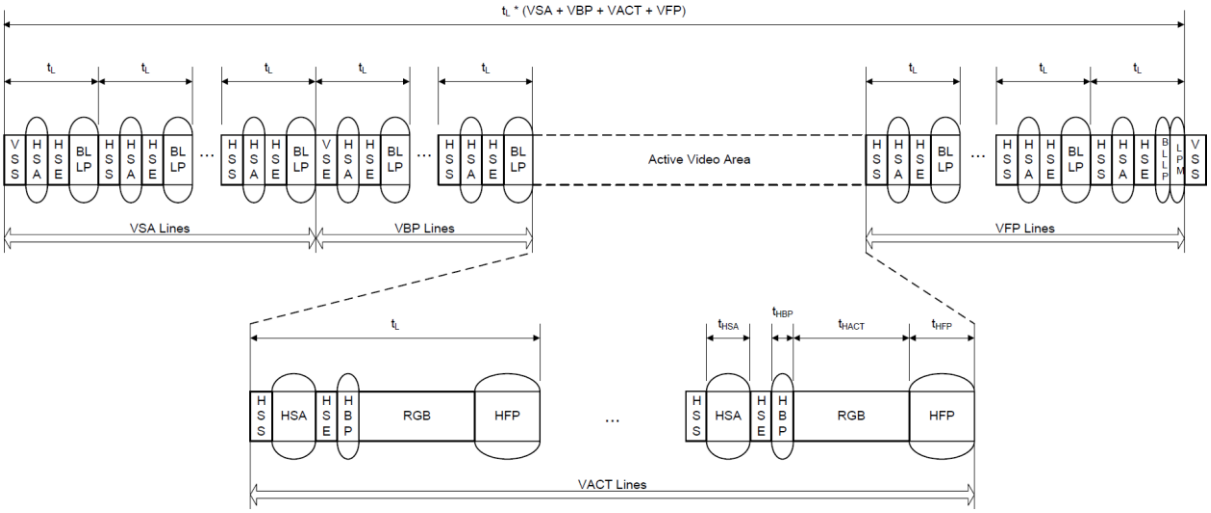


Figure 3-55 Video Mode Interface Timing: Non-burst Mode with Sync Pulses

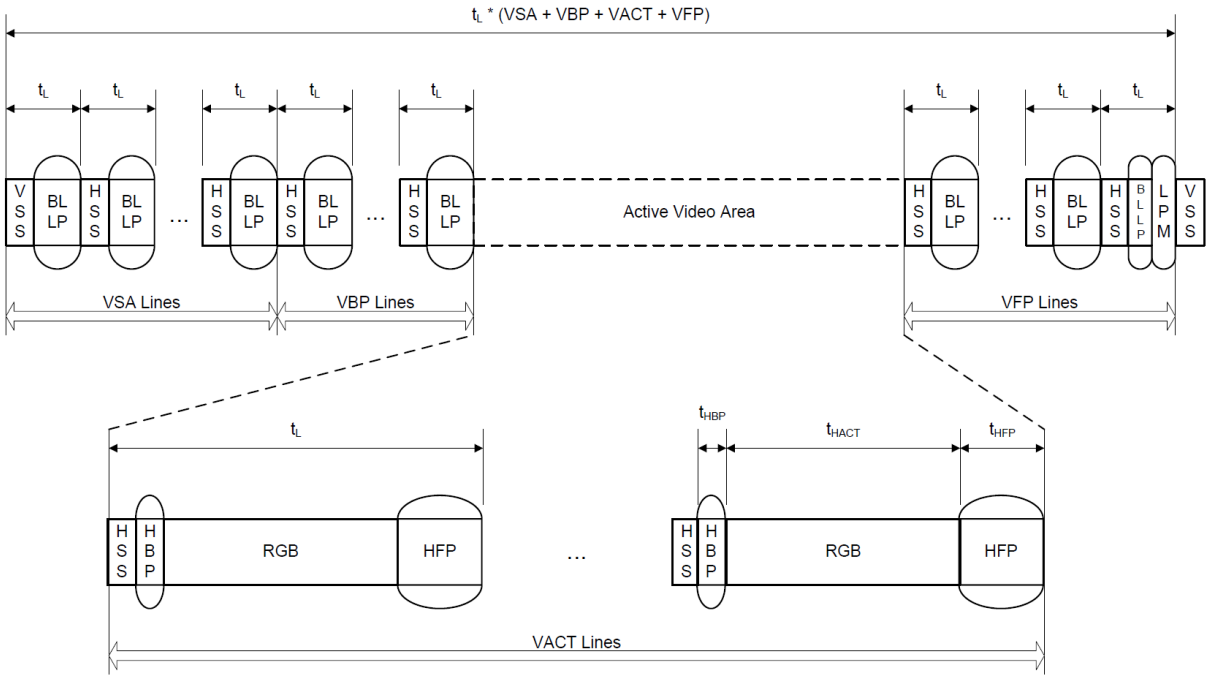


Figure 3-56 Video Mode Interface Timing: Non-burst Mode with Sync Events

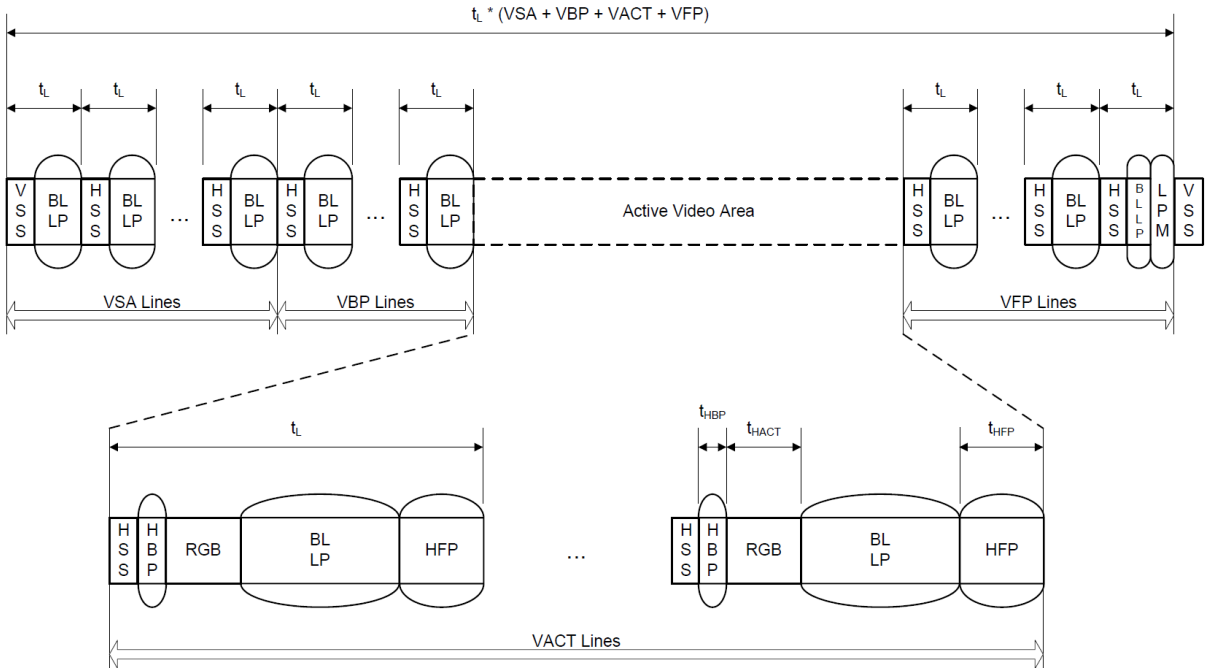


Figure 3-57 Video Mode Interface Timing: Burst Mode

3.8.2.6 Programming Guide

3.8.2.6.1 Initialization

Figure 3-58 illustrates the waveform of the initialization sequence to enable DSI analog block and power up DSI analog block's PLL. Follow the steps described in Table 3-46 to achieve the initialization sequence.

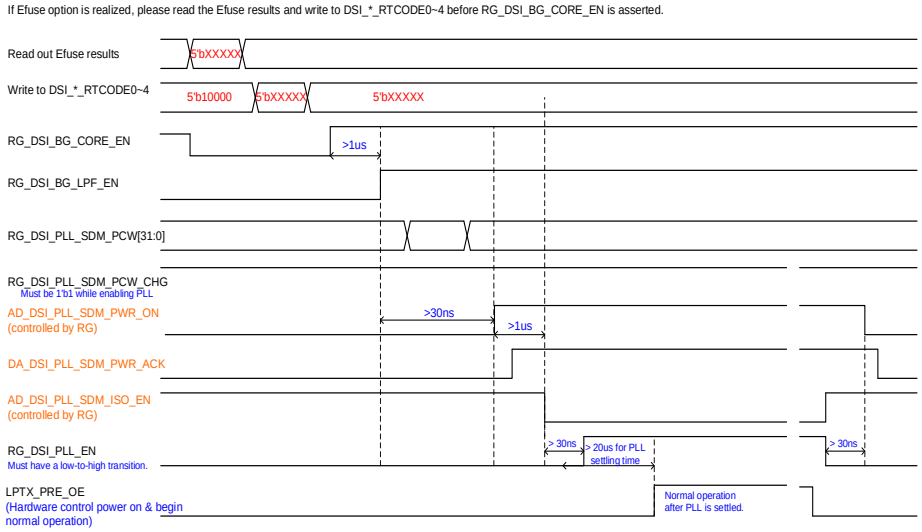


Figure 3-58 Timing Chart of Initialization Sequence

Table 3-46 MIPI D-PHY Initialization Programming Sequence

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Read out eFuse result and write it to impedance RG	W	DSI_*_RTCODE0~4		0 x *00[0]~ 0 x *24[0] *: 1~5

Step	Sequence	R/W	REG_Name	REG_Value	Address
2	Lane CK mode configuration	W	DSI_*_CKMODE_MODE	1	0 x *28[0] *: 1~5
3	Enable BG core power	W	RG_DSI_BG_CORE_EN	1	0 x 00c[7]
4	Wait for 1 µs				
5	Enable BG LP filter	W	RG_DSI_BG_LPF_EN	1	0 x 00c[6]
6	Wait for 30 ns				
7	Enable PLL power	W	AD_DSI_PLL_SDM_PWR_ON	1	0 x 028[0]
8	Enable PLL power isolation	W	AD_DSI_PLL_SDM_ISO_EN	1	0 x 028[1]
9	Wait for 1 µs				
10	Disable PLL isolation	W	AD_DSI_PLL_SDM_ISO_EN	0	0 x 028[1]
11	Wait for 30 ns				
12	Set SDM_PCW by data rate	W	RG_DSI_PLL_SDM_PCW	-	0 x 02C
13	Set POST_DIV by data rate	W	RG_DSI_PLL_POSDIV	-	0 x 030[10:8]
14	Wait for 1 µs				
15	Enable PLL	W	RG_DSI_PLL_EN	1	0 x 030[4]
16	Wait for 20 µs				
17	DSI analog block initialization is done.				

Note:

- The addresses with an underline are MIPI_TX_Config REG, and the addresses without an underline are DSI REG.

3.8.2.6.2 Power Off Sequence

If DSI goes into suspend mode or power saving mode, you can power off the DSI analog block. Figure 3-59 illustrates the waveform of the power-off sequence to disable the DSI analog block and power off the DSI analog block's PLL. The power-off programming steps are listed in Table 3-47. To avoid unexpected power consumption, make sure that all registers are in the default settings.

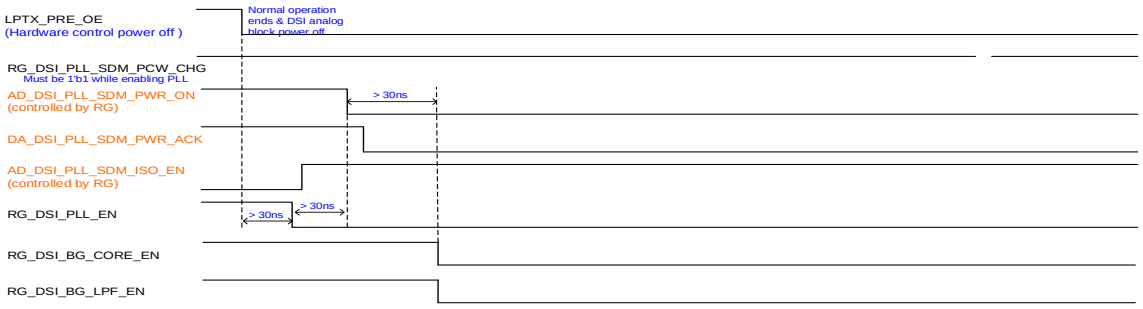


Figure 3-59 Timing Chart of Power-off Sequence

Table 3-47 D-PHY Power-off Programming Sequence

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Turn off all data lanes	W	LANE_NUM	0 x 0	0 x 018[5:2]
2	Wait for 30 ns				
3	Disable PLL	W	RG_DSI_PLL_EN	0	0 x 030[4]
4	Enable PLL isolation	W	AD_DSI_PLL_SDM_ISO_EN	1	0 x 028[1]
5	Wait for 30 ns				

Step	Sequence	R/W	REG_Name	REG_Value	Address
6	Disable PLL power	W	AD_DSI_PLL_SDM_PWR_ON	0	0 x 028[0]
7	Wait for 30 ns				
8	Disable BG LP filter	W	RG_DSI_BG_LPF_EN	0	0 x 00c[6]
9	Disable BG core power	W	RG_DSI_BG_CORE_EN	0	0 x 00c[7]
10	DSI analog block power off done				

Note:

- The addresses with an underline are MIPI_TX_Config REG, and the addresses without an underline are DSI REG.

3.8.2.6.3 Timing Control

3.8.2.6.3.1 PHY Timing Control

All of the timing parameters defined in the MIPI specification should be properly set in the DSI controller programmable registers for correct timing control. The written value is based on the DSI internal clock cycle period, which is related to DSI analog block PLL clock settings through MIPI TX Config engine. And the written value cannot be zero.

For example, the D-PHY timing parameter $T_{HS\text{-}PREPARE}$ must be between 40 ns + 4 * UI and 85 ns + 6 * UI, where UI means time interval, which is equal to the duration of any HS state on clock lane. If the clock lane is set to 500 MHz frequency, as well as bit-rate 1 Gbps, the UI should be 1 ns. In other words, the value of $T_{HS\text{-}PREPARE}$ must between 44 and 91 ns. The internal DSI clock is 8x divided by data rate, as well as 125 MHz. To satisfy $T_{HS\text{-}PREPARE}$, the register value DA_HS_PREP should be 6 to 11 (see Table 3-48).

Table 3-48 D-PHY Timing Parameters Register Setting

	Timing Specification	Absolute Time for UI: 1 ns	DA_HS_PREP Value
$T_{HS\text{-}PREPARE}$	40 ns + 4 * UI~85 ns + 6 * UI	44 to 91 ns	6 to 11

3.8.2.6.3.2 D-PHY PHY Timing Control

Table 3-49 lists the D-PHY timing parameters that should be configured in the DSI registers. Note that for different bit-rate requirements, the UI values vary. For more precise timing control, select DSI internal clock as fast as possible. However, the faster DSI internal clock is set, the more power is wasted. A suitable clock is beneficial to the optimization of system power consumption.

Table 3-49 D-PHY Global Operation Timing Parameter Defined by MIPI Specification

Parameter	Description	Min	Max	Unit
$T_{CLK\text{-}MISS}$	Timeout for receiver to detect absence of Clock transitions and disable the Clock Lane HS-RX.		60	ns
$T_{CLK\text{-}POST}$	Time that the transmitter continues to send HS clock after the last associated Data Lane has transitioned to LP Mode. Interval is defined as the period from the end of $T_{HS\text{-}TRAIL}$ to the beginning of $T_{CLK\text{-}TRAIL}$.	60 ns + 52*UI		ns
$T_{CLK\text{-}PRE}$	Time that the HS clock shall be driven by the transmitter prior to any associated Data Lane beginning the transition from LP to HS mode.	8		UI
$T_{CLK\text{-}PREPARE}$	Time that the transmitter drives the Clock Lane LP-00 Line state immediately before the HS-0 Line state starting the HS transmission.	38	95	ns
$T_{CLK\text{-}SETTLE}$	Time interval during which the HS receiver should ignore any Clock Lane HS transitions, starting from the beginning of $T_{CLK\text{-}PREPARE}$.	95	300	ns

Parameter	Description	Min	Max	Unit
$T_{CLK-TERM-EN}$	Time for the Clock Lane receiver to enable the HS line termination, starting from the time point when Dn crosses $V_{IL,MAX}$.	Time for Dn to reach $V_{TERM-EN}$	38	ns
$T_{CLK-TRAIL}$	Time that the transmitter drives the HS-0 state after the last payload clock bit of a HS transmission burst.	60		ns
$T_{CLK-PREPARE}$ + $T_{CLK-ZERO}$	$T_{CLK-PREPARE}$ + time that the transmitter drives the HS-0 state prior to starting the Clock.	300		ns
$T_{D-TERM-EN}$	Time for the Data Lane receiver to enable the HS line termination, starting from the time point when Dn crosses $V_{IL,MAX}$.	Time for Dn to reach $V_{TERM-EN}$	35 ns + 4*UI	
T_{EOT}	Transmitted time interval from the start of $T_{HS-TRAIL}$ or $T_{CLK-TRAIL}$, to the start of the LP-11 state following a HS burst.		105 ns + n*12*UI	
$T_{HS-EXIT}$	Time that the transmitter drives LP-11 following a HS burst.	100		ns
$T_{HS-PREPARE}$	Time that the transmitter drives the Data Lane LP-00 Line state immediately before the HS-0 Line state starting the HS transmission	40 ns + 4*UI	85 ns + 6*UI	ns
$T_{HS-PREPARE}$ + $T_{HS-ZERO}$	$T_{HS-PREPARE}$ + time that the transmitter drives the HS-0 state prior to transmitting the Sync sequence.	145 ns + 10*UI		ns
$T_{HS-SETTLE}$	Time interval during which the HS receiver shall ignore any Data Lane HS transitions, starting from the beginning of $T_{HS-PREPARE}$. The HS receiver shall ignore any Data Lane transitions before the minimum value, and the HS receiver shall respond to any Data Lane transitions after the maximum value.	85 ns + 6*UI	145 ns + 10*UI	ns
$T_{HS-SKIP}$	Time interval during which the HS-RX should ignore any transitions on the Data Lane, following a HS burst. The end point of the interval is defined as the beginning of the LP-11 state following the HS burst.	40	55 ns + 4*UI	ns
$T_{HS-TRAIL}$	Time that the transmitter drives the flipped differential state after last payload data bit of a HS transmission burst	max(n*8*UI, 60 ns + n*4*UI)		ns
T_{INIT}	Initialization period	100		µs
T_{LPX}	Transmitted length of any Low-Power state period	50		ns
Ratio T_{LPX}	Ratio of $T_{LPX(MASTER)}$/$T_{LPX(SLAVE)}$ between Master and Slave sides	2/3	3/2	
T_{TA-GET}	Time that the new transmitter drives the Bridge state (LP-00) after accepting control during a Link Turnaround.	5*T_{LPX}		ns
T_{TA-GO}	Time that the transmitter drives the Bridge state (LP-00) before releasing control during a Link Turnaround.	4*T_{LPX}		ns
$T_{TA-SURE}$	Time that the new transmitter waits after the LP-10 state before transmitting the Bridge state (LP-00) during a Link Turnaround.	1*T_{LPX}	2*T_{LPX}	ns
T_{WAKEUP}	Time that a transmitter drives a Mark-1 state prior to a Stop state in order to initiate an exit from ULPS.	1		ms

Where n = 1 for Forward-direction HS mode and n = 4 for Reverse-direction HS mode.

The registers of D-PHY timing parameters for data lanes and clock lane are illustrated in Figure 3-60 and Figure 3-61 respectively. The registers for BTA (Bus turn-around) timing are illustrated in Figure 3-62. The unit of them is the clock period of DSI internal clock.

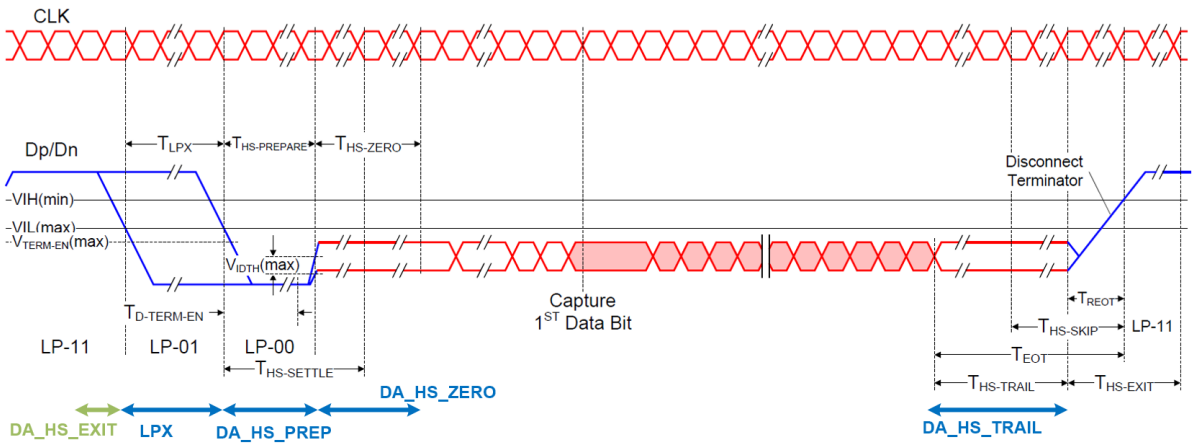


Figure 3-60 Registers for Data Lane Timing Parameters

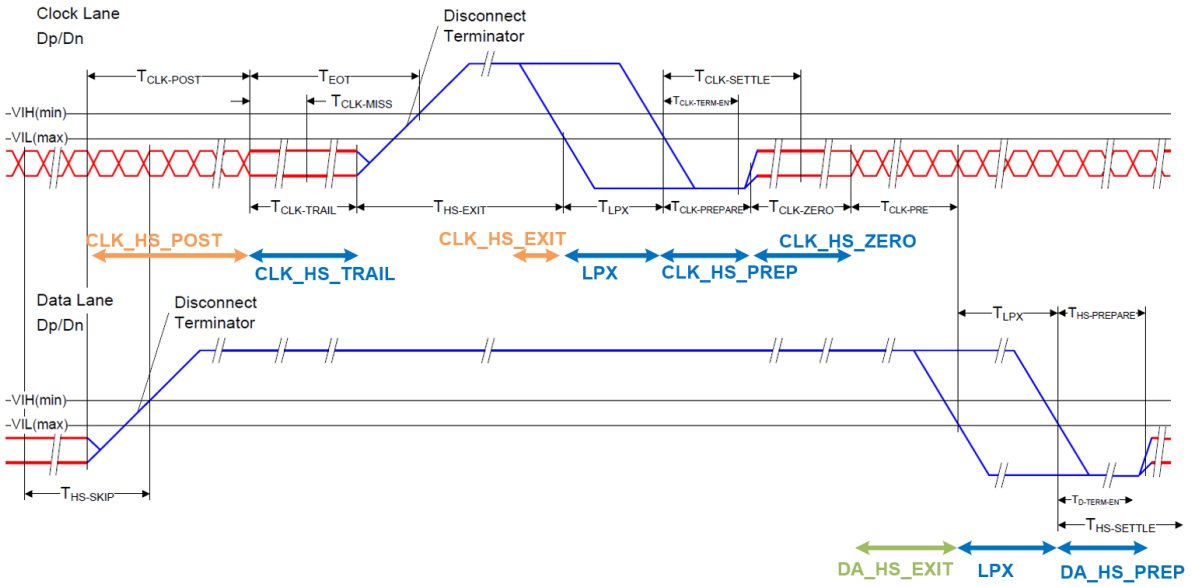


Figure 3-61 Registers for Clock Lane Timing Parameters

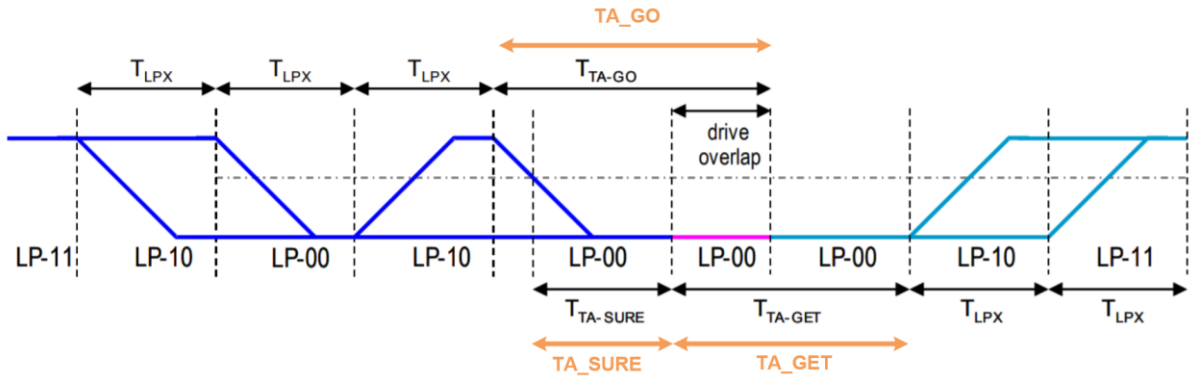


Figure 3-62 Register for BTA Timing Parameters

3.8.2.6.3.3 C-PHY PHY Timing Control

Table 3-50 lists the C-PHY timing parameters that should be configured in the DSI registers. Note that for different symbol-rate requirements, the UI values vary. For example, the UI should be 1 ns when the symbol rate is set to 1Gsps. The internal DSI clock is 7x divided by the symbol rate, as well as 142.8571 MHz.

Table 3-50 C-PHY Global Operation Timing Parameter Defined by MIPI Specification

Parameter	Description	Min	Max	Unit
$T_{3\text{-PREPARE}}$	Time that the transmitter drivers the 3-wire LP-000 line state immediately before the HS_+x line state starting the HS transmission.	38	95	ns
$T_{3\text{-PREBEGIN}}$	The length of the first part of the preamble. $T_{3\text{-PREBEGIN}}$ should be adjustable at the transmitter from 7 UI minimum to 448 UI maximum in increments of 7 UI.	7	448	UI
$T_{3\text{-PROGSEQ}}$	The length of the programmable sequence section of the preamble. The length of $T_{3\text{-PROGSEQ}}$ can be configured by CPHY_PROGSEQ_SKIP_EN (0: $T_{3\text{-PROGSEQ}}$ = 14 UI, 1: $T_{3\text{-PROGSEQ}}$ = 0 UI).	0 or 14		UI
$T_{3\text{-PREEND}}$	The length of the end of the preamble.	7		UI
$T_{3\text{-SYNC}}$	The length of the Sync Word.	7		UI
$T_{3\text{-POST}}$	The length of the Post sequence at the end of the burst. $T_{3\text{-POST}}$ should be adjustable from 7 UI minimum to 224 UI maximum in increments of 7 UI.	7	224	UI
T_{HS_EXIT}	Time that the transmitter drives LP-111 following a HS burst.	100		ns
T_{LPX}	Transmitted length of any low-power state period	50		ns
$T_{A\text{-GET}}$	Time that the new transmitter drives the bridge state (LP-000) after accepting control during a link turnaround	$5*T_{LPX}$		ns
$T_{TA\text{-GO}}$	Time that the transmitter drives the bridge state (LP-000) before releasing control during a link turnaround.	$4*T_{LPX}$		ns
$T_{TA\text{-SURE}}$	Time that the new transmitter waits after the LP-100 state before transmitting the bridge state (LP-000) during a link turnaround.	T_{LPX}	$2*T_{LPX}$	ns
T_{WAKEUP}	Time that a transmitter drives a Mark-1 state prior to a Stop state in order to initiate an exit from ULPS.	1		ms

The registers of C-PHY timing parameters for data lanes and clock lane are illustrated in Figure 3-63. The registers for BTA (Bus Turnaround) timing are illustrated in Figure 3-62. The unit of them is the clock period of DSI internal clock.

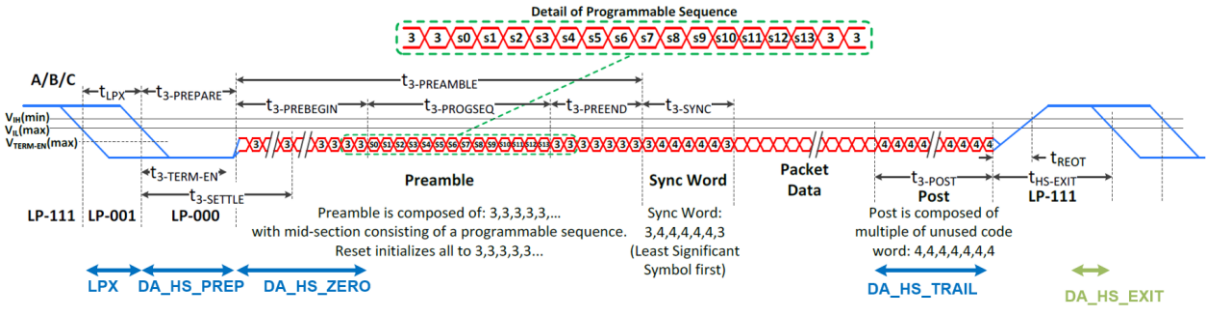


Figure 3-63 Registers for C-PHY Timing Parameters

3.8.2.6.3.4 Video Mode Timing

DSI supports video mode traffic sequences, including sync pulse mode, sync event mode and burst mode. To facilitate the translation of the parameters of packets, see the timing diagrams below for the corresponding register settings. The DSI register of VSA_NL, VBP_NL, VACT_NL and VFP_NL are used to control the line numbers, and the unit is number of line. The ranges of HSA (HPW), HBP, and HFP are specified by DDIC, and the unit is pixel. They are controlled by the DSI register of DSI_HSA_WC, DSI_HBP_WC and DSI_HFP_WC. The unit of DSI_*_WC is byte (8 bits).

3.8.2.6.3.4.1 Non-Burst with Sync-Pulse Mode

A non-burst sync-pulse mode enables the peripheral to accurately reconstruct original video timing, including sync pulse widths. A timing diagram for sync-pulse mode is shown in Figure 3-64, which also shows registers mappings to lines of VSA/VBP/VACT/VFP periods.

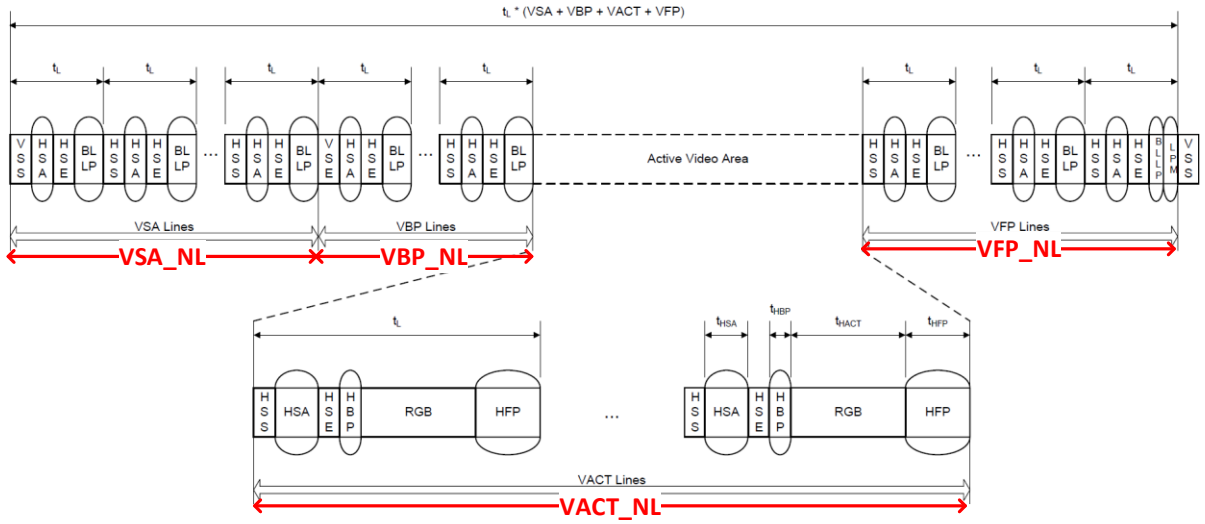


Figure 3-64 Non-burst Transmission: Sync-pulse Mode

3.8.2.6.3.4.2 Non-Burst with Sync-Event Mode

A non-burst sync-event mode is similar to the pulse-sync mode, but accurate reconstruction of sync pulse widths is not required. Therefore, a single sync event is substituted. The timing diagram is shown in Figure 3-65.

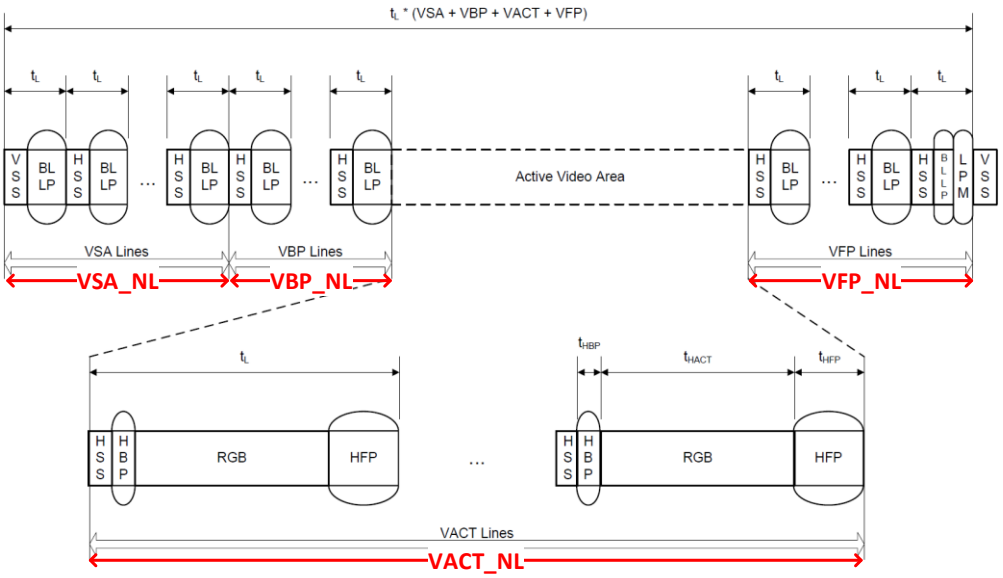


Figure 3-65 Non-burst Transmission: Sync-event Mode

3.8.2.6.3.4.3 Burst Mode

A burst mode allows RGB pixel packets to be time-compressed, leaving more time during a scan line for LP mode to save power or for multiplexing other transmissions onto DSI link. The timing diagram is shown in Figure 3-66.

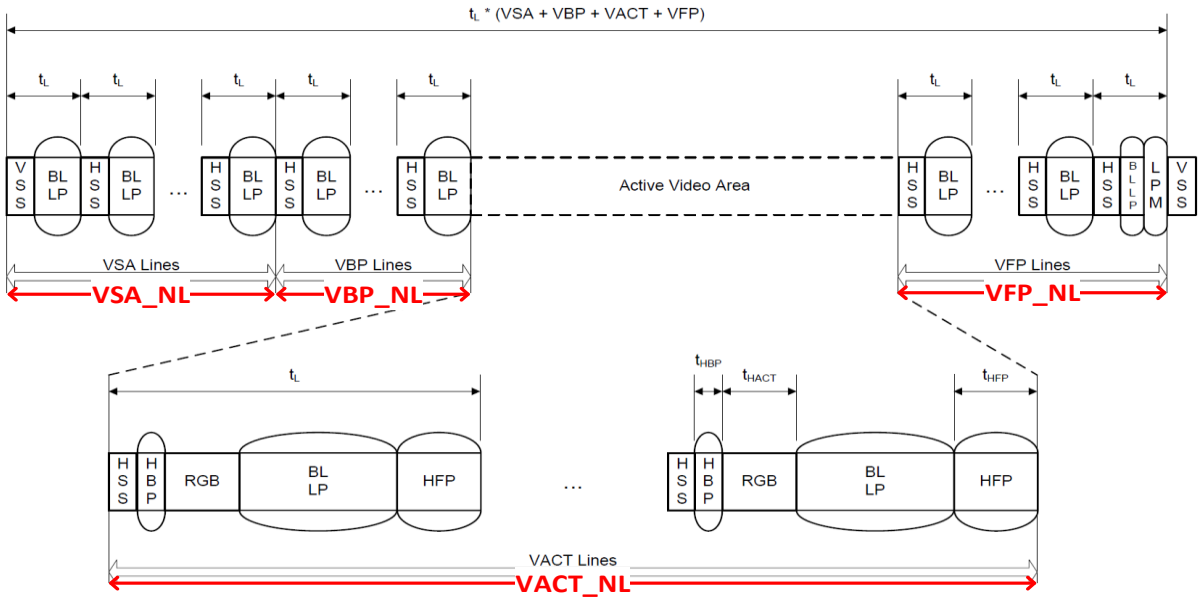


Figure 3-66 Burst Mode Transmission

3.8.2.6.4 Command Mode Packet Transmission

DSI supports command mode transmission through writing commands to a dedicated command queue. By configuring commands and triggering, the transmission can be executed sequentially.

3.8.2.6.4.1 Command Queue

DSI has a dedicated command queue that is 32-bit in width and up to 128-entry in depth, as shown in Figure 3-67. To simplify the settings for transmitting a packet in the command mode, the command queue is designed to categorize all possible transmission types and commands into four primary instructions and unifies all DSI specification commands into one or several 32-bit-wide instructions. Figure 3-67 also illustrates a 32-bit instruction structure with the instruction format of CONFG byte.

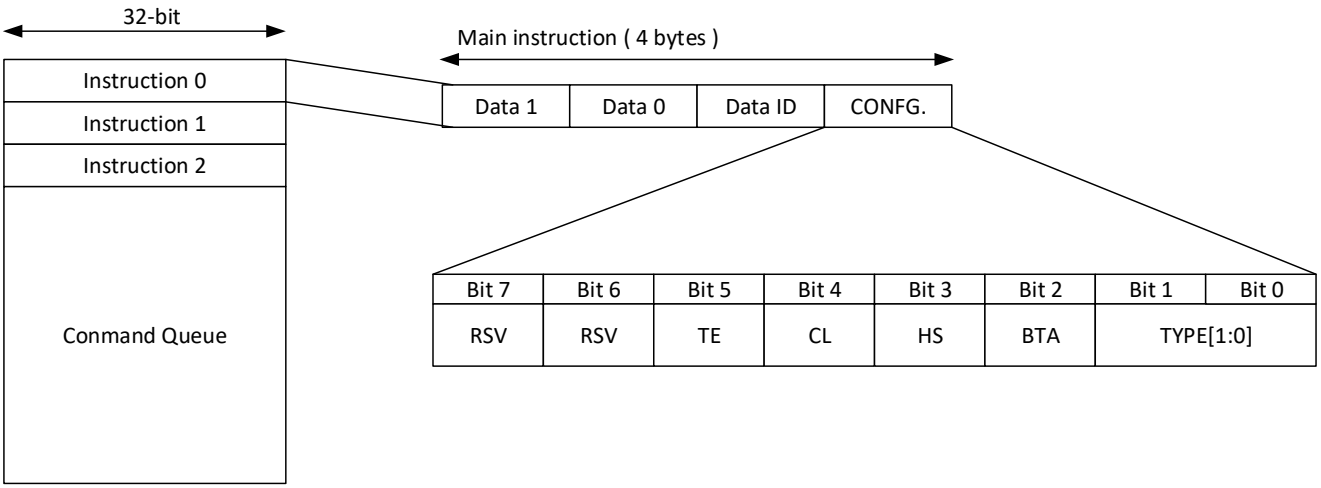


Figure 3-67 DSI Command Queue Instruction Format

Table 3-51 shows the descriptions of the CONFIG byte to an instruction.

Table 3-51 Config Field Description of Main Instruction

REG_Name	REG_Value	Function	Description
Type[1:0]	00	Type 0	Used for DSI short packet read/write command
	01	Type 1	Used for DSI frame buffer write command (long packet)
	10	Type 2	Used for DSI generic long packet write command
	11	Type 3	Used for DSI frame buffer read command (short packet)
BTA	0	Off	Turn around the DSI link after the DSI command is transmitted
	1	On	
HS	0	Off	Enable HS TX transmission for this packet; otherwise, transmit packet via LP TX.
	1	On	
CL	0	8-bit	Select command length for frame buffer read/write instruction. Only effective for type 1 and type 3 instructions.
	1	16-bt	
TE	0	Off	Enable TE request. Will only turn around the DSI link without any packet transmission.
	1	On	
Resv	-	-	Reserved for further use
Resv	-	-	Reserved for further use

3.8.2.6.4.2 Type-0 Instruction

Type-0 instruction is used to transmit short packets. Table 3-52 lists the formats of type-0 instruction where (Data ID + Data 0 + Data 1) is constructed by a DSI short packet command (without ECC).

Table 3-52 Type-0 Instruction Format

Byte 3	Byte 2	Byte 1	Byte 0
Data 1	Data 0	Data ID	CONFG.

To send “Turn on Peripheral” and “Color Mode On” commands, which are transmitted via LP TX and HS TX respectively, request slave response after the second command is finished, and translate the descriptions into two 32-bit instructions, see the steps illustrated in Table 3-53.

Table 3-53 Type-0 TX Example

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Fill command queue entry-0	W	DSI_CMDQ[0]	0 x 0000120C	0 x 200
2	Fill command queue entry-1	W	DSI_CMDQ[1]	0 x 00003200	0 x 204
3	Set command count	W	CMDQ_SIZE	0 x 2	0 x 060[7:0]
4	Start command	W	DSI_START	0 x 1	0 x 000[0]
5	Issue interrupt and receive slave response	R	LPRX_RD_RDY_INT_FLAG	0 x 1	0 x 00C[0]
6	Clear interrupt status	W	LPRX_RD_RDY_INT_FLAG	0 x 1	0 x 00C[0]
7	Read trigger status (Acknowledge)	R	RX_TRIG_0 RX_TRIG_1 RX_TRIG_2 RX_TRIG_3	0 x 4	0 x 088[3:0]
8	Respond read ack to module and go to next commands in queue	W	RACK	0 x 1	0 x 084[0]
9	Issue interrupt and command done	R	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]
10	Clear interrupt status	W	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]

3.8.2.6.4.3 Type-1 Instruction

Type-1 command is used to write data into the frame buffer. As shown in Table 3-54, there are 4 bytes constructing this type of instruction where Mem_start_0 and Mem_start_1 can be generic commands defined by slave vendors or DCS commands. Mem_start_1 is optional, i.e. the memory start/continue command can be single-byte as DCS defines. It depends on the CL bit of the CONFG. The byte indicates whether the DSI controller sends Mem_start_1 or not.

Since the length of frame butter to be updated is not constant, this type of instruction may send several long packets to the slave. The payload data and length of each packet (excluding mem_start_0 and mem_start_1) are prepared by the RDMA controller that couples the output of image data path or layer overlay result to the DSI controller. For the first packet, mem_start_0 and mem_start_1 (if CL = 1) are used as the parameters to inform the slave that the host is starting to write the frame buffer. For the remaining packets, the register value DSI_RWMEM_CONTI[15:0] will be used as the parameters to inform the slave side to write these data following the last pixel of the previous packet. For more flexibility, Mem_start_0, Mem_start_1, DSI_RWMEM_CONTI[15:0] and CL are all programmable.

You need to set up two registers to define the packet length and packet count for a frame-based type-1 transmission. Frame width in bytes should be set to PS_WC, and frame height in lines should be set to DSI_VACT_NL, respectively. These two registers are used in both video and command mode frame data transmission.

Table 3-54 Type-1 Instruction Format

Byte 3	Byte 2	Byte 1	Byte 0
Mem start 1 (optional)	Mem start 0	Data ID	CONFG.

Refer to the example in Table 3-55 to write the frame buffer via DCS commands in the HS TX mode.

Table 3-55 Type-1 TX Example

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Fill command queue entry-0	W	DSI_CMDQ[0]	0 x 002C3909	0 x 200
2	Set command count	W	CMDQ_SIZE	0 x 1	0 x 060[7:0]
3	Set memory continue command	W	DSI_RWMEM_CONTI	0 x 3C	0 x 090[15:0]
4	Start command	W	DSI_START	0 x 1	0 x 000[0]
5	Issue interrupt and command done	R	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]
6	Clear interrupt status	W	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]

3.8.2.6.4.4 Type-2 Instruction

Type-2 instruction is used to send a long packet. As shown in Table 3-56, this type of main instruction requires several sub-instructions that do not have CONFIG. To send a type-2 command, write a CONFIG with TYPE = 2 and packet header information (Data ID + 2 byte word count) to entry 0, and write a series of data bytes in size of word count to the following entries, excluding ECC and checksum. The bytes in the following entries will be treated as long packet data instead of the next instruction until the word count size is reached. The command queue count should be set as the number of multiple entries used.

The type-2 command is sent in LPTX mode due to memory latency in reading sub-instruction data. Besides, type-2 command should be sent individually without the next instruction followed. For the 32-entry command queue, the maximum word count for long packet is 124 bytes.

Table 3-56 Type-2 Instruction Format

Byte 3	Byte 2	Byte 1	Byte 0
WC 1	WC 0	Data ID	CONFG.
Data 3	Data 2	Data 1	Data 0
		Data WC-1	Data WC-2

See Table 3-57 for the example of sending three parameters (0 x 33, 0 x 22, 0 x 11) by a generic long packet command with 3-byte word count.

Table 3-57 Type-2 TX Example

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Fill command queue entry-0 (data ID and word count)	W	DSI_CMDQ[0]	0 x 00032902	0 x 200
2	Fill command queue entry-1 (parameters)	W	DSI_CMDQ[1]	0 x 00112233	0 x 204

Step	Sequence	R/W	REG_Name	REG_Value	Address
3	Set command count	W	CMDQ_SIZE	0 x 2	0 x 060[7:0]
4	Start command	W	DSI_START	0 x 1	0 x 000[0]
5	Issue interrupt and command done	R	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]
6	Clear interrupt status	W	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]

Note that the RPT bit of CONFIG.is designed for this type of instruction. It is useful for the NULL packet or blanking packet. For example, if a null packet is to be sent, only the main instruction (entry-0 of command queue) will be needed, and the following payload data will be sent as “0”.

3.8.2.6.4.5 Type-3 Instruction

Type-3 instruction is used for reading frame buffer. As shown in Table 3-58, the format is the same as that of type-1. When this instruction is executed, the host will first send a short packet with memory start parameter given in byte 2 and byte 3 and automatically issue the next packet by memory continuous parameters programmed in DSI_RWMEM_CONTI[15:0]. The number of total packets required to be sent depends on the DSI_FRM_BC and “maximum return packet size”. For example, to read 1,024 bytes from the frame buffer in the slave, and the “maximum return packet size” is set to “4”, there will be another 255 short packets with memory continuous parameters to be sent successively after the first short packet described in main instruction is sent.

Table 3-58 Type-3 Instruction Format

Byte 3	Byte 2	Byte 1	Byte 0
Men start 1 (optional)	Mem start 0	Data ID	CONFG.

See Table 3-59 for the example of using type-3 instruction to perform the frame buffer read.

Table 3-59 Type-3 TX Example

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Fill command queue entry-0	W	DSI_CMDQ[0]	0 x 002E0603	0 x 200
2	Set command count	W	CMDQ_SIZE	0 x 1	0 x 060[7:0]
3	Set memory read continue command	W	DSI_RWMEM_CONTI	0 x 3E	0 x 090[15:0]
4	Start command	W	DSI_START	0 x 1	0 x 000[0]
5	Issue interrupt and receive slave response	R	LPRX_RD_RDY_INT_FLAG	0 x 1	0 x 00C[0]
6	Read receive data bytes	R	BYTE0 BYTE1 BYTE2 BYTE3	-	0 x 074
7	Clear interrupt status	W	LPRX_RD_RDY_INT_FLAG	0 x 1	0 x 00C[0]
8	Respond with read acknowledgement to module and go to next commands in queue	W	RACK	0 x 1	0 x 084[0]
9	Issue interrupt and command done	R	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]
10	Clear interrupt status	W	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]

3.8.2.6.5 Tearing Effect Detection

3.8.2.6.5.1 Peripheral TE

DSI has the ability to receive peripheral Tearing Effect (TE) signals via BTA process. The TE signal is LPTX transmitted signal from panel. Before starting to receive TE signals from the peripheral, make sure a DCS command of “set_tear_on” is sent and register configuration of TE is enabled in the peripheral to avoid TE hanged issue. Here is an example in Table 3-60 to show how to trigger TE commands by command queue.

Table 3-60 Example of Peripheral TE Signaling Detection

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Fill command queue: DCS “set_tear_on”	W	DSI_CMDQ[0]	0 x 00351500	0 x 200
2	Fill command queue: Get TE	W	DSI_CMDQ[1]	0 x 00000020	0 x 204
3	Fill command queue: Type-1 command	W	DSI_CMDQ[2]	0 x 002C3909	0 x 208
4	Set command count	W	CMDQ_SIZE	0 x 3	0 x 060[7:0]
5	Set memory write continue command	W	DSI_RWMEM_CONTI	0 x 3C	0 x 090[15:0]
6	Start command	W	DSI_START	0 x 1	0 x 000[0]
7	Issue interrupt and receive TE	R	TE_RDY_INT_FLAG	0 x 1	0 x 00C[2]
8	Read trigger status (TE)	R	RX_TRIG_0, RX_TRIG_1, RX_TRIG_2, RX_TRIG_3	0 x 2	0 x 088[3:0]
9	Clear interrupt status	W	TE_RDY_INT_FLAG	0 x 1	0 x 00C[2]
10	Response read ack to module and go to next commands in queue	W	RACK	0 x 1	0 x 084[0]
11	Issue interrupt and command done	R	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]
12	Clear interrupt status	W	CMD_DONE_INT_FLAG	0 x 1	0 x 00C[1]

3.8.2.6.5.2 External TE

In certain cases, an external TE pin may be used instead of TE signals for some reasons. DSI supports such mechanism to issue external TE interrupt signals. Refer to the sequences shown in Table 3-61 for detailed control information.

Table 3-61 Example of External TE Pin Detection

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Enable external TE interrupt	W	TE_RDY_INT_EN	0 x 1	0 x 008[2]
2	Select external TE polarity (active-high)	W	EXT_TE_EDGE_SEL	0 x 0	0 x 018[10]
3	Enable external TE	W	EXT_TE_EN	0 x 1	0 x 018[9]
4	Issue interrupt and receive external TE	R	TE_RDY_INT_FLAG	0 x 1	0 x 00C[2]
5	Clear interrupt status	W	TE_RDY_INT_FLAG	0 x 1	0 x 00C[2]

3.8.2.6.6 Switch of Video Mode and Command Mode

The switch between video mode and command mode is controlled by MODE_CON (DSI base address + 0 x 014[1:0]). And such switch is allowed only when DSI is not busy. DSI will hang up if the mode is switched when DSI is still busy. There are three ways to know if DSI is non-busy, which are shown in Table 3-62, Table 3-63 and Table 3-64.

Table 3-62 DSI Non-busy Detection by DSI_BUSY Register

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Stop DSI	W	DSI_START	0 x 0	0 x 000[0]
2	Wait for DSI non-busy	R	DSI_BUSY	0 x 0	0 x 00C[31]
3	Switch mode	W	MODE_CON	-	0 x 014[1:0]
4	Mode switch done				

Table 3-63 DSI Non-busy Detection by DSI_DONE Event

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Stop DSI	W	DSI_START	0 x 0	0 x 000[0]
2	Wait for DSI_DONE event				
3	Switch mode	W	MODE_CON	-	0 x 014[1:0]
4	Mode switch done				

Table 3-64 DSI Non-busy Detection by VM_DONE/CM_DONE Interrupt

Step	Sequence	R/W	REG_Name	REG_Value	Address
1	Enable VM_DONE interrupt (video mode)/CM_DONE interrupt (command mode)	W	VM_DONE_INT_EN/ CM_DONE_INT_EN	0 x 1	0 x 008[3]/[1]
2	Stop DSI	W	DSI_START	0 x 0	0 x 000[0]
3	Wait for VM_DONE (video mode)/CM_DONE IRQ (command mode)				
4	Switch mode	W	MODE_CON	-	0 x 014[1:0]
5	Mode switch done				

3.8.2.6.7 Lane/Trio Swap Function

3.8.2.6.7.1 D-PHY Lane Swap

The D-PHY lane swap function can swap the source of all lanes. The source of each lane can be chosen by MIPI_TX_*_SEL, which is shown in Table 3-65. The schematic diagram of lane swap mux is shown in Figure 3-68.

Table 3-65 D-PHY Lane Swap Control Register

REG_Name	Description	Default REG_Value	Address
MIPI_TX_PHY2_SEL	Lane2 LP data P selection	4'd0	MIPI_TX_Config base address + 0 x 040[7:4]
MIPI_TX_CPHY0BC_SEL	Lane2 LP data N selection	4'd1	MIPI_TX_Config base address + 0 x 040[11:8]
MIPI_TX_PHY0_SEL	Lane0 LP data P selection	4'd2	MIPI_TX_Config base address + 0 x 040[15:12]
MIPI_TX_CPHY1AB_SEL	Lane0 LP data N selection	4'd3	MIPI_TX_Config base address + 0 x 040[19:16]
MIPI_TX_PHYC_SEL	Lanec LP data P selection	4'd4	MIPI_TX_Config base address + 0 x 040[23:20]
MIPI_TX_CPHY1CA_SEL	Lanec LP data N selection	4'd5	MIPI_TX_Config base address + 0 x 040[27:24]

REG_Name	Description	Default REG_Value	Address
MIPI_TX_PHY1_SEL	Lane1 LP data P selection	4'd6	MIPI_TX_Config base address + 0 x 040[31:28]
MIPI_TX_CPHY2BC_SEL	Lane1 LP data N selection	4'd7	MIPI_TX_Config base address + 0 x 044[3:0]
MIPI_TX_PHY3_SEL	Lane3 LP data P selection	4'd8	MIPI_TX_Config base address + 0 x 044[7:4]
MIPI_TX_CPHYXXX_SEL	Lane3 LP data N selection	4'd9	MIPI_TX_Config base address + 0 x 044[11:8]
MIPI_TX_PHY2_HSDATA_SEL	Lane2 HS data selection	4'd0	MIPI_TX_Config base address + 0 x 048[3:0]
MIPI_TX_PHY0_HSDATA_SEL	Lane0 HS data selection	4'd2	MIPI_TX_Config base address + 0 x 048[11:8]
MIPI_TX_PHYC_HSDATA_SEL	LaneC HS data selection	4'd4	MIPI_TX_Config base address + 0 x 048[19:16]
MIPI_TX_PHY1_HSDATA_SEL	Lane1 HS data selection	4'd6	MIPI_TX_Config base address + 0 x 048[27:24]
MIPI_TX_PHY3_HSDATA_SEL	Lane3 HS data selection	4'd8	MIPI_TX_Config base address + 0 x 04C[3:0]

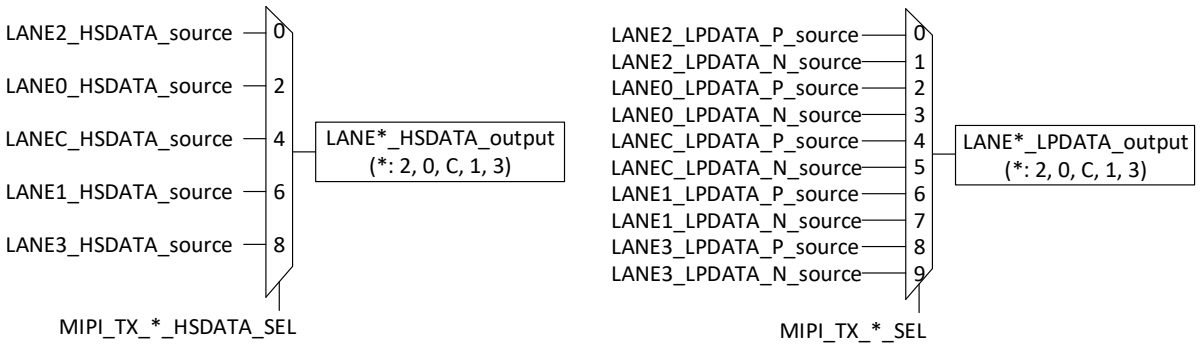


Figure 3-68 D-PHY Lane Swap Mux

3.8.2.6.7.2 C-PHY Trio Swap

The C-PHY trio swap function can swap the source of all trios. The source of each trio can be chosen by MIPI_TX_*_SEL, which is shown in Table 3-66. The schematic diagram of trio swap mux is shown in Figure 3-69.

Table 3-66 C-PHY Trio Swap Control Register

REG_Name	Description	Default REG_Value	Address
MIPI_TX_PHY2_SEL	T0A LP data selection	4'd0	MIPI_TX_Config base address + 0 x 040[7:4]
MIPI_TX_CPHY0BC_SEL	T0B LP data selection	4'd1	MIPI_TX_Config base address + 0 x 040[11:8]
MIPI_TX_PHY0_SEL	T0C LP data selection	4'd2	MIPI_TX_Config base address + 0 x 040[15:12]
MIPI_TX_CPHY1AB_SEL	T1A LP data selection	4'd3	MIPI_TX_Config base address + 0 x 040[19:16]
MIPI_TX_PHYC_SEL	T1B LP data selection	4'd4	MIPI_TX_Config base address + 0 x 040[23:20]
MIPI_TX_CPHY1CA_SEL	T1C LP data selection	4'd5	MIPI_TX_Config base address + 0 x 040[27:24]
MIPI_TX_PHY1_SEL	T2A LP data selection	4'd6	MIPI_TX_Config base address + 0 x 040[31:28]
MIPI_TX_CPHY2BC_SEL	T2B LP data selection	4'd7	MIPI_TX_Config base address + 0 x 044[3:0]
MIPI_TX_PHY3_SEL	T2C LP data selection	4'd8	MIPI_TX_Config base address + 0 x 044[7:4]
MIPI_TX_PHY2_HSDATA_SEL	T0A HS data selection	4'd0	MIPI_TX_Config base address + 0 x 048[3:0]
MIPI_TX_CPHY0BC_HSDATA_SEL	T0B HS data selection	4'd1	MIPI_TX_Config base address + 0 x 048[7:4]

REG_Name	Description	Default REG_Value	Address
MIPI_TX_PHY0_HSDATA _SEL	T0C HS data selection	4'd2	MIPI_TX_Config base address + 0 x 048[11:8]
MIPI_TX_CPHY1AB_HSDATA _SEL	T1A HS data selection	4'd3	MIPI_TX_Config base address + 0 x 048[15:12]
MIPI_TX_PHYC_HSDATA _SEL	T1B HS data selection	4'd4	MIPI_TX_Config base address + 0 x 048[19:16]
MIPI_TX_CPHY1CA_HSDATA _SEL	T1C HS data selection	4'd5	MIPI_TX_Config base address + 0 x 048[23:20]
MIPI_TX_PHY1_HSDATA _SEL	T2A HS data selection	4'd6	MIPI_TX_Config base address + 0 x 048[27:24]
MIPI_TX_CPHY2BC_HSDATA _SEL	T2B HS data selection	4'd7	MIPI_TX_Config base address + 0 x 048[31:28]
MIPI_TX_PHY3_HSDATA _SEL	T2C HS data selection	4'd8	MIPI_TX_Config base address + 0 x 04C[3:0]
MIPI_TX_CPHY0_HS_SEL	T0 HS ctrl signal selection	2'd0	MIPI_TX_Config base address + 0 x 044[25:24]
MIPI_TX_CPHY1_HS_SEL	T0 HS ctrl signal selection	2'd1	MIPI_TX_Config base address + 0 x 044[27:26]
MIPI_TX_CPHY2_HS_SEL	T0 HS ctrl signal selection	2'd2	MIPI_TX_Config base address + 0 x 044[29:28]

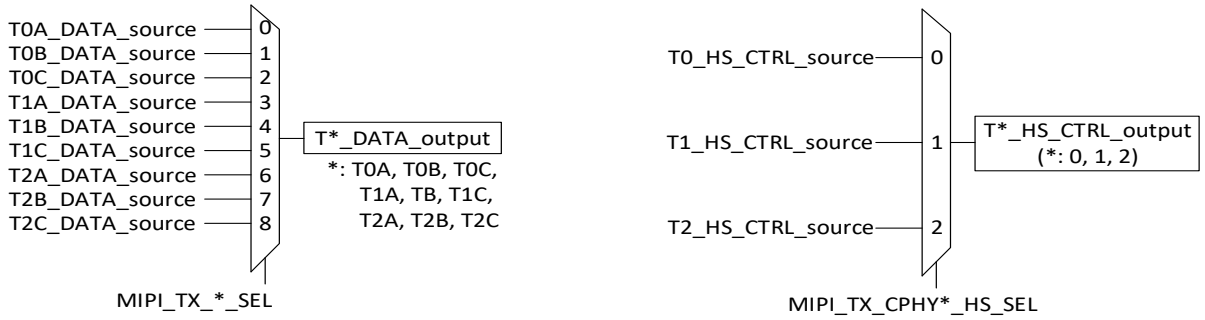


Figure 3-69 C-PHY Trio Swap Mux

3.8.2.7 DSI Signal Descriptions

Table 3-67 presents DSI signal descriptions.

Table 3-67 DSI Signal Descriptions

Signal Name	Type	Description		Ball Location
DSI1		**D-PHY Mode**	**C-PHY Mode**	
DSI1_CKN_T1C	AIO	DSI1 clock lane (negative)	DSI1 Trio1 C	AR3
DSI1_CKP_T1B	AIO	DSI1 clock lane (positive)	DSI1 Trio1 B	AR2
DSI1_D0N_T1A	AIO	DSI1 data lane 0 (negative)	DSI1 Trio1 A	AR1
DSI1_D0P_T0C	AIO	DSI1 data lane 0 (positive)	DSI1 Trio0 C	AP2
DSI1_D1N_T2B	AIO	DSI1 data lane 1 (negative)	DSI1 Trio2 B	AU4
DSI1_D1P_T2A	AIO	DSI1 data lane 1 (positive)	DSI1 Trio2 A	AU3
DSI1_D2N_T0B	AO	DSI1 data lane 2 (negative)	DSI1 Trio0 B	AN1
DSI1_D2P_T0A	AIO	DSI1 data lane 2 (positive)	DSI1 Trio0 A	AN2
DSI1_D3N	AIO	DSI1 data lane 3 (negative)	-	AR4
DSI1_D3P_T2C	AIO	DSI1 data lane 3 (positive)	DSI1 Trio2 C	AT4

Signal Name	Type	Description		Ball Location
DSI1_TE	DI	DSI1 tearing effect control	DSI1 tearing effect control	AM8, AT9
DSI0		**D-PHY Mode**	**C-PHY Mode**	
DSI0_CKN_T1C	AIO	DSI0 clock lane (negative)	DSI0 Trio1 C	AL2
DSI0_CKP_T1B	AIO	DSI0 clock lane (positive)	DSI0 Trio1 B	AM2
DSI0_D0N_T1A	AIO	DSI0 data lane 0 (negative)	DSI0 Trio1 A	AK3
DSI0_D0P_T0C	AIO	DSI0 data lane 0 (positive)	DSI0 Trio0 C	AK4
DSI0_D1N_T2B	AIO	DSI0 data lane 1 (negative)	DSI0 Trio2 B	AM3
DSI0_D1P_T2A	AIO	DSI0 data lane 1 (positive)	DSI0 Trio2 A	AL3
DSI0_D2N_T0B	AIO	DSI0 data lane 2 (negative)	DSI0 Trio0 B	AJ4
DSI0_D2P_T0A	AIO	DSI0 data lane 2 (positive)	DSI0 Trio0 A	AJ3
DSI0_D3N	AIO	DSI0 data lane 3 (negative)	-	AN4
DSI0_D3P_T2C	AIO	DSI0 data lane 3 (positive)	DSI0 Trio2 C	AM4
DSI_TE	DI	DSI0 tearing effect control	DSI0 tearing effect control	AT10, AT11

3.8.3 Display Digital Parallel Interface (DPI)

3.8.3.1 Overview

The device includes two DPI controllers, DPI0 and DPI1, which output digital video data and timing signals. DPI0 is used to directly interface an external display panel, while DPI1 provides data and timings to HDMITX module.

3.8.3.2 Features

Each DPI controller supports the following key features:

- Flexible output data bus width and formats:
  - DPI0 up to 16-bit bus: RGB565/YUV422 8-bit
  - DPI1 up to 30-bit bus: RGB565/RGB 8-bit, 10bit/YUV444 8-bit, 10-bit/YUV422 8-bit, 10-bit, 12-bit
- Resolution up to 1920 × 1080 @ 60fps (for DPI0 only)
- Fixed-coefficient color space conversion
- Embedded synchronization timings for BT.656-like output format
- Dual edge output format
- YUV444 to YUV422 chroma down-sampling
- Internal pattern generator

3.8.3.3 DPI Signal Descriptions

Table 3-68 presents DPI signal descriptions.

Table 3-68 DPI Signal Descriptions

Signal Name	Type	Description	Ball Location
DPI_CK	DO	DPI pixel clock	AT14
DPI_D0	DO	DPI data 0	AN14

Signal Name	Type	Description	Ball Location
DPI_D1	DO	DPI data 1	AM14
DPI_D10	DO	DPI data 10	AL12
DPI_D11	DO	DPI data 11	AK12
DPI_D12	DO	DPI data 12	AP12
DPI_D13	DO	DPI data 13	AL13
DPI_D14	DO	DPI data 14	AP13
DPI_D15	DO	DPI data 15	AN12
DPI_D2	DO	DPI data 2	AL14
DPI_D3	DO	DPI data 3	AK14
DPI_D4	DO	DPI data 4	AU15
DPI_D5	DO	DPI data 5	AR14
DPI_D6	DO	DPI data 6	AK11
DPI_D7	DO	DPI data 7	AK10
DPI_D8	DO	DPI data 8	AP14
DPI_D9	DO	DPI data 9	AR12
DPI_DE	DO	DPI data enable	AT12
DPI_HSYNC	DO	DPI horizontal synchronization	AU14
DPI_VSYNC	DO	DPI vertical synchronization	AT13

3.8.3.4 DPI Timing Characteristics

Table 3-69 and Figure 3-70 present timing characteristics for DPI in the device.

Table 3-69 DPI Timing Characteristics

No.	Parameter		Min	Max	Unit
DPI01	t_c	Cycle time	6.73(1)		ns
DPI02	D	Duty cycle, DPI_CK	45	55	%
DPI03	t_{RISE}	Rise time		1.374	ns
DPI04	t_{FALL}	Fall time		1.374	ns
DPI05	t_d	Delay time, other signals to DPI_CK	1.683		ns

(1) For maximum operating clock frequency, refer to Table 6-1.

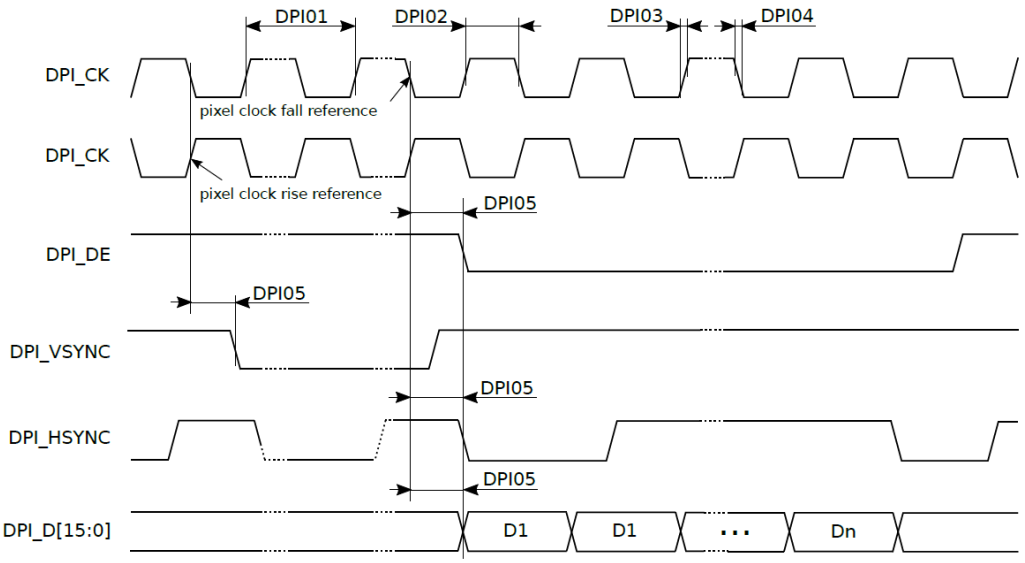


Figure 3-70 DPI Timing Diagram

3.8.3.5 DPI Block Diagram

- Timing generator (tgen): Generate timing signals like vsync, hsync, de.
- Pattern generator (pat): Generate internal pattern. If pattern is enabled, input data will be replaced by internal pattern data.
- FIFO controller (afifo): Turn 1T1P or 1T2P to 1T1P and turn mm_clk to dpi_pixel_clk.
- Matrix: Perform color space conversion and support 19 internal matrices as well as programmable matrix.
- Chroma low pass filter (clpf) and YUV422: Down sample YUV444 to YUV422.
- Embedded sync (embsync): Embed sync signals to data channel and support BT656-like output format.
- Outstage: Support channel swap and DDR, output video data and timing.

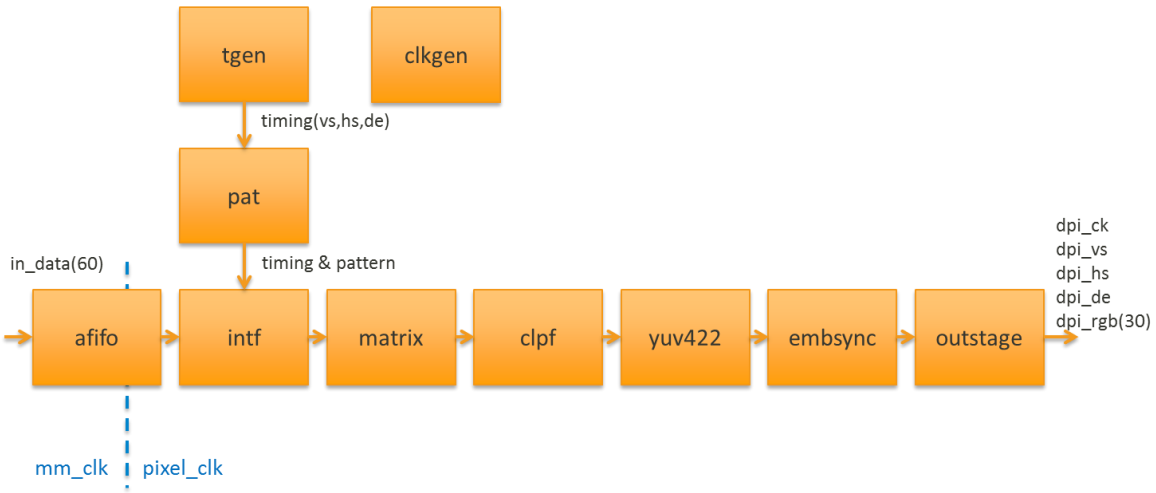


Figure 3-71 DPI Block Diagram

3.8.3.6 DPI Programming Guide

The following figure shows the DPI programming flow diagram. Firstly, configure each timing register based on the target frame timing. Then, reset and enable DPI.

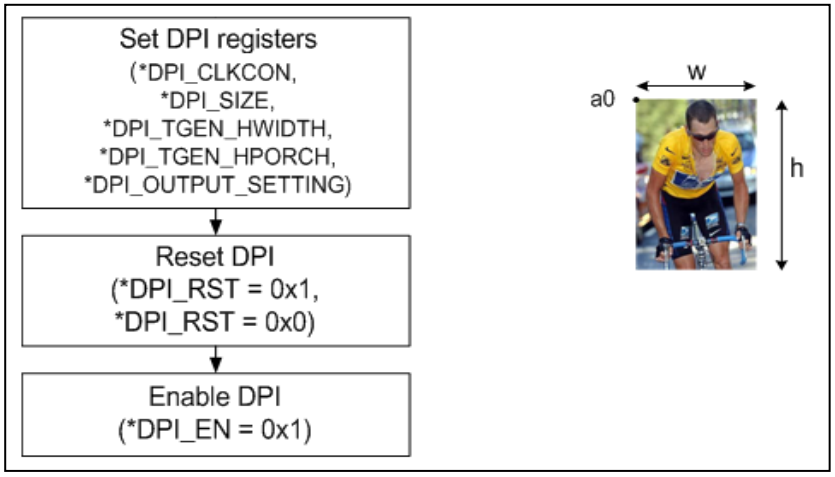


Figure 3-72 Programming Flow Diagram

3.8.4 Embedded DisplayPort Interface (EDPTX)

3.8.4.1 Overview

The EDPTX provides the electrical transport for video and auxiliary data between the device and an external display module. The communication link is handled through the eDP Auxiliary Channel (EDPAUX).

3.8.4.2 Features

The EDPTX supports the following key features:

- Compliance with eDP v1.4 standard
- Single link output port with 4× main lanes
- Up to 5.4 Gbps per lane (HBR2)
- Hot-Plug Detect (HPD) line
- Auxiliary channel lane:
  - Manchester-II coding
  - Clock extracted from the data stream
  - 1 Mbps bit rate
- Input data formats: RGB 8-bit/10-bit, YUV444 8-bit/10-bit, YUV422 8-bit/10-bit
- RGB444 8-bit/10-bit color format
- 8-bit to 10-bit encoder
- Alternative Scrambler Seed Reset (ASSR) function
- Inter-lane skew function
- Up to 4K2K @ 60 Hz resolution (10-bit, HBR2 without DSC)

3.8.4.3 EDPTX Signal Descriptions

Table 3-70 presents EDPTX signal descriptions.

Table 3-70 EDPTX Signal Descriptions

Signal Name	Type	Description	Ball Location
EDP_LN0_TXN	AIO	EDPTX lane 0 (negative)	AD33
EDP_LN0_TXP	AIO	EDPTX lane 0 (positive)	AD34
EDP_LN1_TXN	AIO	EDPTX lane 1 (negative)	AD37
EDP_LN1_TXP	AIO	EDPTX lane 1 (positive)	AD36
EDP_LN2_TXN	AIO	EDPTX lane 2 (negative)	AF36
EDP_LN2_TXP	AIO	EDPTX lane 2 (positive)	AE35
EDP_LN3_TXN	AIO	EDPTX lane 3 (negative)	AF33
EDP_LN3_TXP	AIO	EDPTX lane 3 (positive)	AF34
EDP_TX_HPD	DI	EDPTX hot plug detect	H34, AN10
EDPAUXN	AIO	EDPTX auxiliary channel (negative)	AE31
EDPAUXP	AIO	EDPTX auxiliary channel (positive)	AE32

3.8.4.4 Block Diagram

The EDP_TX_MAC block is the main design. It supports HBR2 solution. Source video signals can come from GPU. The AUX can be controlled by the software.

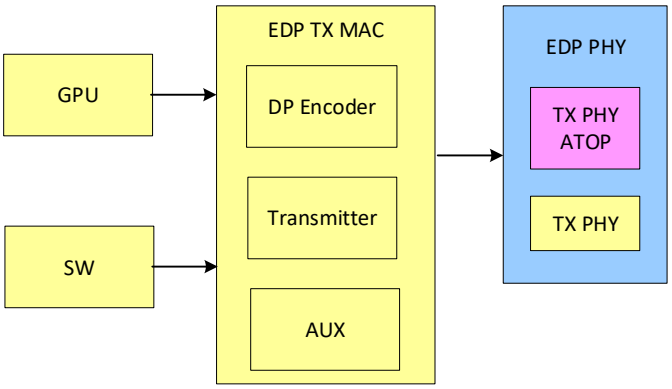


Figure 3-73 EDP TX Block Diagram with Other IPs

3.8.4.4.1 EDPTX Encoder

Pixel clock and link clock are the main clocks in DP. Hence, the design is related to the speed limitation. The Transfer Unit (TU) calculator generates the appropriate TU value to avoid TBC FIFO from being empty or full. The lane arbiter splits video data into 4/2/1 DP lanes. The video data packer packages video data into DP format. Video MN GEN circuit calculates the M value, which is inserted to VBID and Main Stream Attribute (MSA) packets. VBID includes the vblank flag, field flag, interlace flag (I mode), video mute flag, audio mute flag, HDCP sync detect and DSC flag. MSA includes video timing information (HV_total, HV_width, etc.). Symbol mixer contains VBID, MSA and video data.

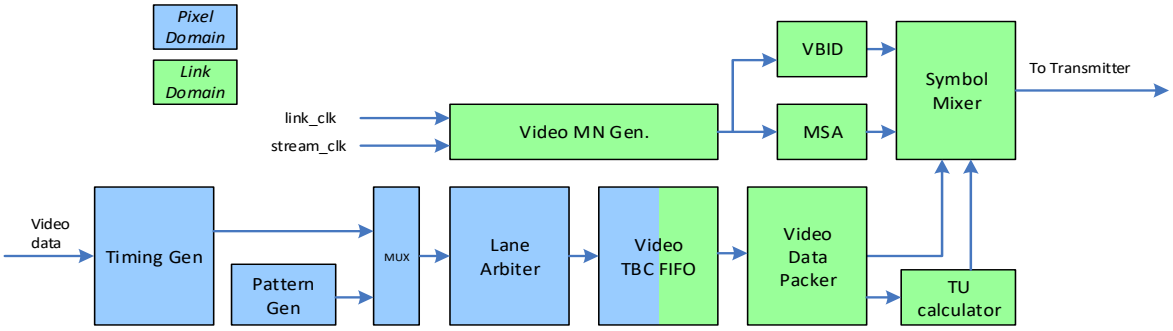


Figure 3-74 EDP TX Encoder Block Diagram

3.8.4.4.2 DPTX Transmitter

1. Main link data path is processed in 4P, clk_ls = link_rate/40.
2. TX Training Control generates TPS1/TPS2/TPS3/TPS4 patterns.
3. Pre-Encode MISC includes lane mux, PN swap, bit reverse and data swap.
4. Post-Encode MISC includes lane mux, PN swap, bit reverse and data swap.
5. Program/PRBS Pattern gen can generate 8/11/80-bit and PRBS test patterns.

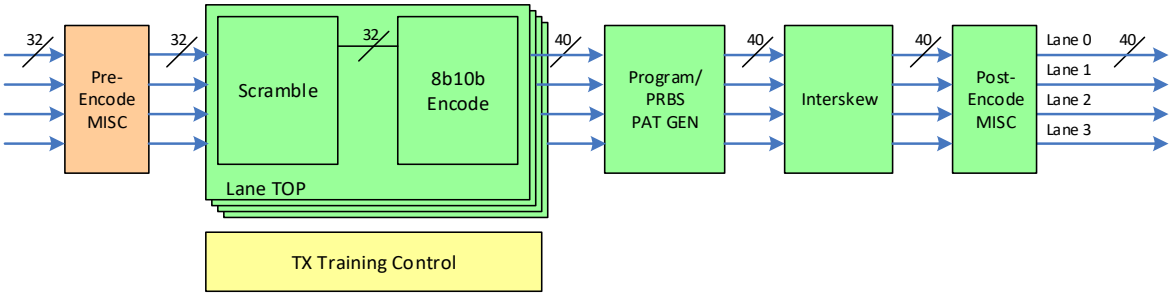


Figure 3-75 DP TX Transmitter Block Diagram

3.8.4.4.3 DPTX AUX

The software controls PHY TX to transmit AUX sequence and receive RX site reply data by PHY RX.

1. PHY TX includes Manchester encoder.
2. PHY RX includes Manchester decoder.

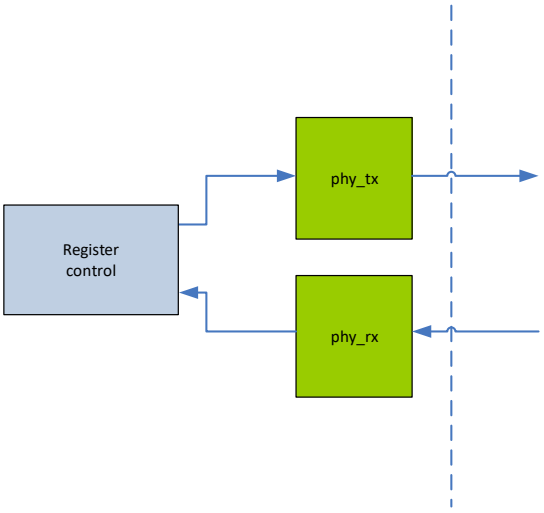


Figure 3-76 DP TX AUX Diagram

3.8.4.5 General SW MAC Setting (APN)

3.8.4.5.1 MSA

dptx_reg_3030_dp_enc_4p[9:0] = 10'h3ff
dptx_reg_3010_dp_enc_4p[15:0]: htotal
dptx_reg_3014_dp_enc_4p[15:0]: vtotal
dptx_reg_3018_dp_enc_4p[15:0]: hstart
dptx_reg_301C_dp_enc_4p[15:0]: vstart
dptx_reg_3020_dp_enc_4p[15:0]: hwidth
dptx_reg_3024_dp_enc_4p[15:0]: vheigth
dptx_reg_3028_dp_enc_4p[14:0]: hsw
dptx_reg_3028_dp_enc_4p[15]: hsp
dptx_reg_302C_dp_enc_4p[14:0]: vsw
dptx_reg_302C_dp_enc_4p[15]: vsp

3.8.4.5.2 Lane Number

Lane number control register exists in encoder and transmitter.

4-lane: dptx_reg_3000_dp_enc_4p[1:0] = 2'b10
dptx_reg_34A4_dp_trans_4p[3:2] = 2'b10
dptx_reg_35F0_dp_trans_4p[3:2] = 2'b10

2-lane: dptx_reg_3000_dp_enc_4p[1:0] = 2'b01
dptx_reg_34A4_dp_trans_4p[3:2] = 2'b01
dptx_reg_35F0_dp_trans_4p[3:2] = 2'b01

1-lane: dptx_reg_3000_dp_enc_4p[1:0] = 2'b00
dptx_reg_34A4_dp_trans_4p[3:2] = 2'b00
dptx_reg_35F0_dp_trans_4p[3:2] = 2'b00

3.8.4.5.3 Training Pattern (1/2/3/4)

Normal mode : dptx_reg_3400_dp_trans_4p[15:12] = 4'b0000
Training Pattern 1: dptx_reg_3400_dp_trans_4p[15:12] = 4'b0001
Training Pattern 2: dptx_reg_3400_dp_trans_4p[15:12] = 4'b0010
Training Pattern 3: dptx_reg_3400_dp_trans_4p[15:12] = 4'b0100
Training Pattern 4: dptx_reg_3400_dp_trans_4p[15:12] = 4'b1000

3.8.4.5.4 CTS Pattern

CP2520 pattern 1: dptx_reg_3478_dp_trans_4p[0]
CP2520 pattern 2: dptx_reg_3478_dp_trans_4p[1]
PRBS pattern: dptx_reg_3444_dp_trans_4p[3]

3.8.4.5.5 Alternate Scrambler Seed Reset (ASSR)

Scrambler Seed Reset = hfffe
dptx_reg_3404_dp_trans_4p[1]

3.8.4.5.6 Enhanced Frame Mode

dptx_reg_3000_dp_enc_4p[4]

3.8.4.5.7 Pre-Lane Mux

dptx_reg_3404_dp_trans_4p[1:0]: Lane0 mux
dptx_reg_3404_dp_trans_4p[3:2]: Lane1 mux
dptx_reg_3404_dp_trans_4p[5:4]: Lane2 mux
dptx_reg_3404_dp_trans_4p[7:6]: Lane3 mux

3.8.4.5.8 Post-Lane Mux

dptx_reg_3408_dp_trans_4p[9:8]: Lane0 mux
dptx_reg_3408_dp_trans_4p[11:10]: Lane1 mux
dptx_reg_3408_dp_trans_4p[13:12]: Lane2 mux
dptx_reg_3408_dp_trans_4p[15:14]: Lane3 mux

3.8.4.5.9 Post-PN Swap

dptx_reg_3406_dp_trans_4p[1]: PN swap

3.8.4.5.10 Initial Setting

RGB 30BPP: dptx_reg_303C_dp_enc_4p[14:8] = 7'h02
RGB 24BPP: dptx_reg_303C_dp_enc_4p[14:8] = 7'h03
RGB 18BPP: dptx_reg_303C_dp_enc_4p[14:8] = 7'h04

MISC0: dptx_reg_3034_dp_enc_4p[7:0], value according to DP specification.
MISC1: dptx_reg_3034_dp_enc_4p[15:8]

3.8.4.5.11 MSA Delay Line Setting

dptx_reg_32F8_dp_enc_4p_2[8]: MSA line delay function enable
dptx_reg_32F8_dp_enc_4p_2[7:0]: Line delay count

3.8.4.5.12 TU Calculation

$$TU\ num = 64 \times \frac{pixel\_bpp \times pixel\_rate}{lane\_num \times link\_rate \times 8} = 64 \times \frac{pixe\_bpp \times M\ value}{lane\_num \times N\ value \times 8}$$

dptx_reg_33C0_dp_enc_4p_2[6:0]: Read hardware TU number

3.8.4.5.13 AUX PHY Initial Setting

dptx_reg_367C_dp_tx_aux[12] = 1'b1
dptx_reg_3670_dp_tx_aux[10] = 1'b0
dptx_reg_3658_dp_tx_aux[10] = 1'b0

3.8.4.5.14 AUX 400us Timeout Threshold

dptx_reg_360C_dp_tx_aux[12:0]: The unit is xtal period.

3.8.4.5.15 DPCD Write

dptx_reg_3644_dp_tx_aux[3:0]: Request command
{dptx_reg_364C_dp_tx_aux, dptx_reg_3648_dp_tx_aux[15:0]}: Request address
dptx_reg_3650_dp_tx_aux[15:12]: Request length
dptx_reg_3634_dp_tx_aux[7:0]: Request write data
dptx_reg_3630_dp_tx_aux[3]: Request ready

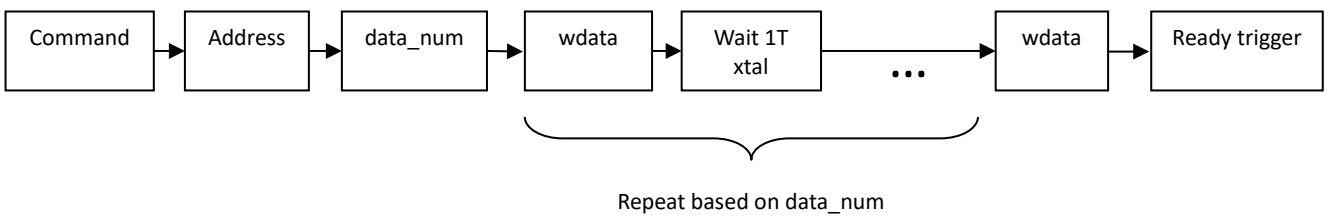
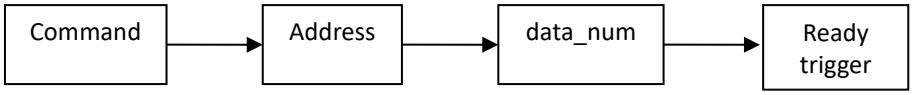


Figure 3-77 DPCD Writing Flow

dptx_reg_3640_dp_tx_aux[6]: Sink reply complete IRQ
dptx_reg_3640_dp_tx_aux[0]: AUX timeout IRQ
dptx_reg_3624_dp_tx_aux[3:0]: Sink reply command

3.8.4.5.16 DPCD Read

These control registers are the same as DPCD write registers.



dptx_reg_3620_dp_tx_aux[8]: The software reads data pulse.

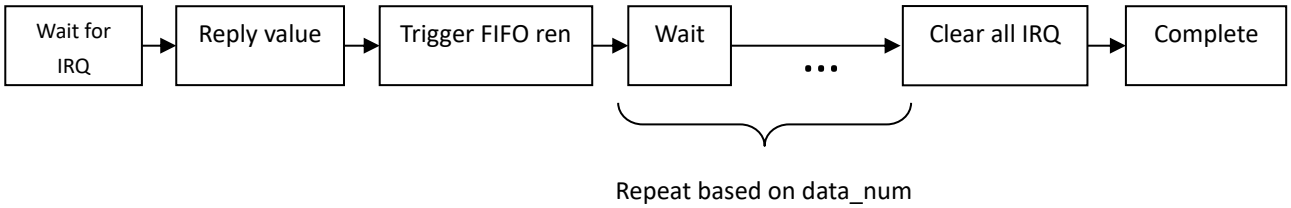


Figure 3-78 DPCD Reading Flow

3.8.4.5.17 I2C Write

The flow is the same as that of DPCD write.

3.8.4.5.18 I2C Write No Length

This flow is different. Since AUX sequence has no length, data_num and wdata can be ignored. And no_length has to be cleared after the communication is completed.

dptx_reg_362C_dp_tx_aux[0]: No length setting

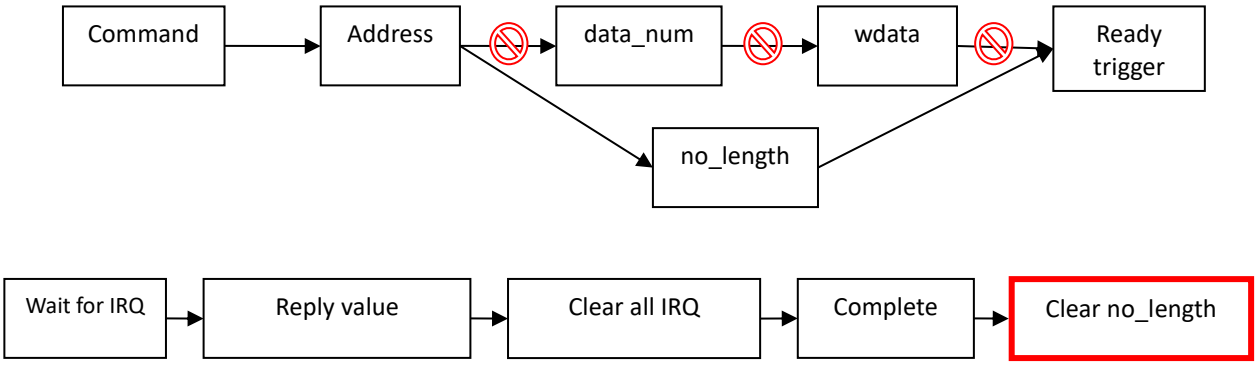


Figure 3-79 I2C Writing Flow

3.8.4.5.19 I2C Read

The flow is the same as that of DPCD read.

3.8.4.5.20 I2C Read No Length

The flow is the same as DPCD write no length, but it is a write command.

3.8.4.5.21 AUX PHY FIFO Status

dptx_reg_368C_dp_tx_aux[0]: RX FIFO done flag
dptx_reg_368C_dp_tx_aux[1]: Clear RX FIFO done flag
dptx_reg_368C_dp_tx_aux[2]: TX FIFO done flag
dptx_reg_368C_dp_tx_aux[3]: Clear TX FIFO done flag

When toggling dptx_reg_3620_dp_tx_aux[8], the software could poll dptx_reg_368C_dp_tx_aux[0]. If this bit is 1'b0, it will read the next data. Before reading the next data, clear the dptx_reg_368C_dp_tx_aux[0] status by dptx_reg_368C_dp_tx_aux[1].

When toggling dptx_reg_3634_dp_tx_aux, the software could poll dptx_reg_368C_dp_tx_aux[2]. If this bit is 1'b0, it will write the next data to this FIFO. Before writing the next data to TX FIFO, clear the dptx_reg_368C_dp_tx_aux[2] status by dptx_reg_368C_dp_tx_aux[3].

3.8.4.5.22 AUX TX Request Pre-charge + Preamble Number

dptx_reg_3630_dp_tx_aux[15:8]: pre-charge + preamble number

3.8.5 DisplayPort Interface (DPTX)

3.8.5.1 Overview

The DPTX provides digital video and auxiliary data transfer between the device and an external display module. The communication link is handled through the DP Auxiliary Channel (DPAUX).

3.8.5.2 Features

The DPTX supports the following key features:

- Compliance with DP v1.4 standard
- Single link output port with 4× main lanes, configured as follows:
  - 4× lanes with up to 8.1 Gbps per lane (HBR3)
- Hot-Plug Detect (HPD) line
- Auxiliary channel lane:
  - Manchester-II coding
  - Clock extracted from the data stream
  - 1 Mbps bit rate
- Input data formats: RGB 8-bit/10-bit, YUV444 8-bit/10-bit, YUV422 8-bit/10-bit
- Output data format: RGB444 8-bit/10-bit
- 8-bit to 10-bit encoder
- Alternative Scrambler Seed Reset (ASSR) function

- Inter-lane skew function
- Up to 4K2K @ 60 Hz resolution (10-bit, HBR2 without DSC)

3.8.5.3 DPTX Signal Descriptions

Table 3-71 presents DPTX signal descriptions.

Table 3-71 DPTX Signal Descriptions

Signal Name	Type	Description	Ball Location
DP_LN0_TXN	AIO	DPTX lane 0 (negative)	AH35
DP_LN0_TXP	AIO	DPTX lane 0 (positive)	AG35
DP_LN1_TXN	AIO	DPTX lane 1 (negative)	AJ37
DP_LN1_TXP	AIO	DPTX lane 1 (positive)	AJ36
DP_LN2_TXN	AIO	DPTX lane 2 (negative)	AK34
DP_LN2_TXP	AIO	DPTX lane 2 (positive)	AJ34
DP_LN3_TXN	AIO	DPTX lane 3 (negative)	AL33
DP_LN3_TXP	AIO	DPTX lane 3 (positive)	AL32
DP_TX_HPD	DI	DPTX hot plug detect	H32, AP25
DPAUXN	AIO	DPTX auxiliary channel (negative)	AG31
DPAUXP	AIO	DPTX auxiliary channel (positive)	AG30

3.8.5.4 Block Diagram

The DP_TX_MAC block is the main design. It supports HBR3 solution. Source video signals can come from GPU. The AUX can be controlled by the software.

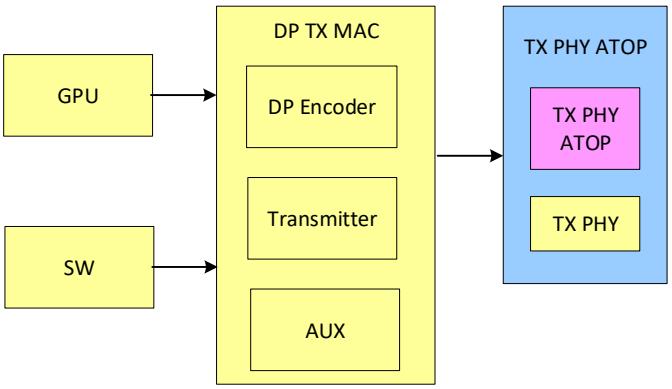


Figure 3-80 DP TX Block Diagram with Other IPs

3.8.5.4.1 DPTX Encoder

Pixel clock and link clock are the main clocks in DP. Hence, the design is related to the speed limitation. The Transfer Unit (TU) calculator generates the appropriate TU value to avoid TBC FIFO from being empty or full. The lane arbiter splits video data into 4/2/1 DP lanes. The video data packer packages video data into DP format. Video MN GEN circuit calculates M value, which is inserted to VBID and Main Stream Attribute (MSA) packet. VBID includes vblank flag, field flag, interlace flag

(I mode), video mute flag, audio mute flag, HDCP sync detect and DSC flag. MSA includes video timing information (HV_total, HV_width, etc.). Symbol mixer contains VBID, MSA and video data.

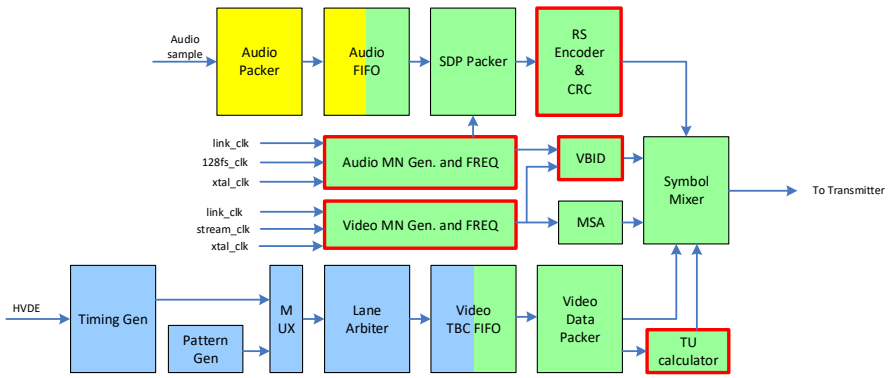


Figure 3-81 DP TX Encoder Block Diagram

3.8.5.4.2 DPTX Transmitter

1. Main link data path is processed in 4P, clk_ls = link_rate/40.
2. TX Training Control generates TPS1/TPS2/TPS3/TPS4 patterns.
3. Pre-Encode MISC includes lane mux, PN swap, bit reverse and data swap.
4. Post-Encode MISC includes lane mux, PN swap, bit reverse and data swap.
5. Program/PRBS Pattern gen can generate 8/11/80-bit and PRBS test patterns.

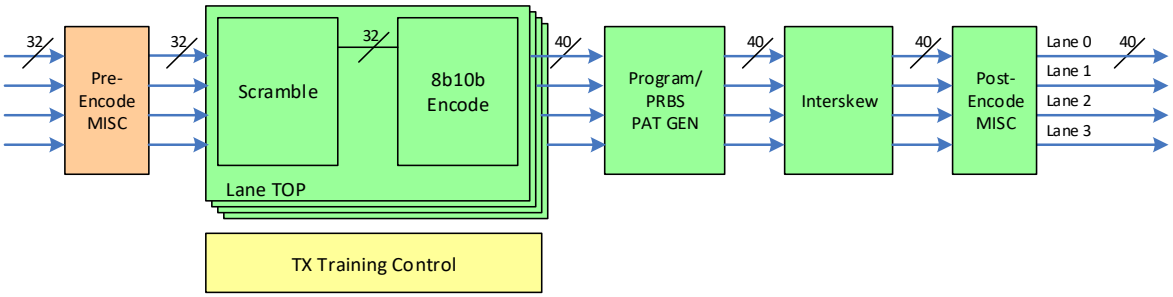


Figure 3-82 DP TX Transmitter Block Diagram

3.8.5.4.3 DP TX AUX

The software controls PHY TX to transmit AUX sequence and receive RX site reply data by PHY RX.

1. PHY TX includes Manchester encoder.
2. PHY RX includes Manchester decoder.

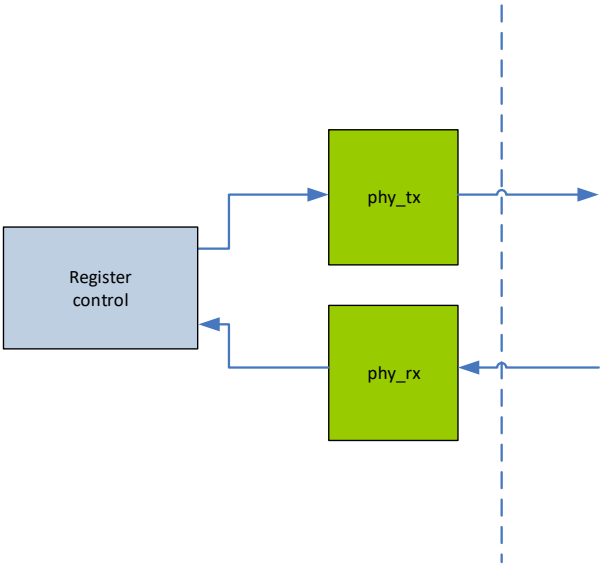


Figure 3-83 DP TX AUX Diagram

3.8.5.5 General SW MAC Setting (APN)

3.8.5.5.1 MSA

dptx_reg_3030_dp_enc_4p[9:0] = 10'h3ff
dptx_reg_3010_dp_enc_4p[15:0]: htotal
dptx_reg_3014_dp_enc_4p[15:0]: vtotal
dptx_reg_3018_dp_enc_4p[15:0]: hstart
dptx_reg_301C_dp_enc_4p[15:0]: vstart
dptx_reg_3020_dp_enc_4p[15:0]: hwidth
dptx_reg_3024_dp_enc_4p[15:0]: vheigth
dptx_reg_3028_dp_enc_4p[14:0]: hsw
dptx_reg_3028_dp_enc_4p[15]: hsp
dptx_reg_302C_dp_enc_4p[14:0]: vsw
dptx_reg_302C_dp_enc_4p[15]: vsp

3.8.5.5.2 Lane Number

Lane number control register exists in encoder and transmitter.

4-lane: dptx_reg_3000_dp_enc_4p[1:0] = 2'b10
dptx_reg_34A4_dp_trans_4p[3:2] = 2'b10
dptx_reg_35F0_dp_trans_4p[3:2] = 2'b10

2-lane: dptx_reg_3000_dp_enc_4p[1:0] = 2'b01
dptx_reg_34A4_dp_trans_4p[3:2] = 2'b01
dptx_reg_35F0_dp_trans_4p[3:2] = 2'b01

1-lane: dptx_reg_3000_dp_enc_4p[1:0] = 2'b00
dptx_reg_34A4_dp_trans_4p[3:2] = 2'b00

dptx_reg_35F0_dp_trans_4p[3:2] = 2'b00

3.8.5.5.3 Training Pattern (1/2/3/4)

Normal mode : dptx_reg_3400_dp_trans_4p[15:12] = 4'b0000
Training Pattern 1: dptx_reg_3400_dp_trans_4p[15:12] = 4'b0001
Training Pattern 2: dptx_reg_3400_dp_trans_4p[15:12] = 4'b0010
Training Pattern 3: dptx_reg_3400_dp_trans_4p[15:12] = 4'b0100
Training Pattern 4: dptx_reg_3400_dp_trans_4p[15:12] = 4'b1000

3.8.5.5.4 CTS Pattern

CP2520 pattern 1: dptx_reg_3478_dp_trans_4p[0]
CP2520 pattern 2: dptx_reg_3478_dp_trans_4p[1]
PRBS pattern: dptx_reg_3444_dp_trans_4p[3]

3.8.5.5.5 Alternate Scrambler Seed Reset (ASSR)

Scrambler Seed Reset = hfffe
dptx_reg_3404_dp_trans_4p[1]

3.8.5.5.6 Enhanced Frame Mode

dptx_reg_3000_dp_enc_4p[4]

3.8.5.5.7 Pre-Lane Mux

dptx_reg_3404_dp_trans_4p[1:0]: Lane0 mux
dptx_reg_3404_dp_trans_4p[3:2]: Lane1 mux
dptx_reg_3404_dp_trans_4p[5:4]: Lane2 mux
dptx_reg_3404_dp_trans_4p[7:6]: Lane3 mux

3.8.5.5.8 Post-Lane Mux

dptx_reg_3408_dp_trans_4p[9:8]: Lane0 mux
dptx_reg_3408_dp_trans_4p[11:10]: Lane1 mux
dptx_reg_3408_dp_trans_4p[13:12]: Lane2 mux
dptx_reg_3408_dp_trans_4p[15:14]: Lane3 mux

3.8.5.5.9 Post-PN Swap

dptx_reg_3406_dp_trans_4p[1]: PN swap

3.8.5.5.10 Initial Setting

RGB 30BPP: dptx_reg_303C_dp_enc_4p[14:8] = 7'h02
RGB 24BPP: dptx_reg_303C_dp_enc_4p[14:8] = 7'h03
RGB 18BPP: dptx_reg_303C_dp_enc_4p[14:8] = 7'h04
MISC0: dptx_reg_3034_dp_enc_4p[7:0], value according to DP specification.
MISC1: dptx_reg_3034_dp_enc_4p[15:8]

3.8.5.5.11 MSA Delay Line Setting

dptx_reg_32F8_dp_enc_4p_2[8]: MSA line delay function enable
dptx_reg_32F8_dp_enc_4p_2[7:0]: Line delay count

3.8.5.5.12 TU Calculation

$$TU\ num = 64 \times \frac{pixel\_bpp \times pixel\_rate}{lane\_num \times link\_rate \times 8} = 64 \times \frac{pixe\_bpp \times M\ value}{lane\_num \times N\ value \times 8}$$

dptx_reg_33C0_dp_enc_4p_2[6:0]: Read hardware TU number

3.8.5.5.13 AUX PHY Initial Setting

dptx_reg_367C_dp_tx_aux[12] = 1'b1
dptx_reg_3670_dp_tx_aux[10] = 1'b0
dptx_reg_3658_dp_tx_aux[10] = 1'b0

3.8.5.5.14 AUX 400us Timeout Threshold

dptx_reg_360C_dp_tx_aux[12:0]: The unit is xtal period.

3.8.5.5.15 DPCD Write

dptx_reg_3644_dp_tx_aux[3:0]: Request command
{dptx_reg_364C_dp_tx_aux, dptx_reg_3648_dp_tx_aux[15:0]}: Request address
dptx_reg_3650_dp_tx_aux[15:12]: Request length
dptx_reg_3634_dp_tx_aux[7:0]: Request write data
dptx_reg_3630_dp_tx_aux[3]: Request ready

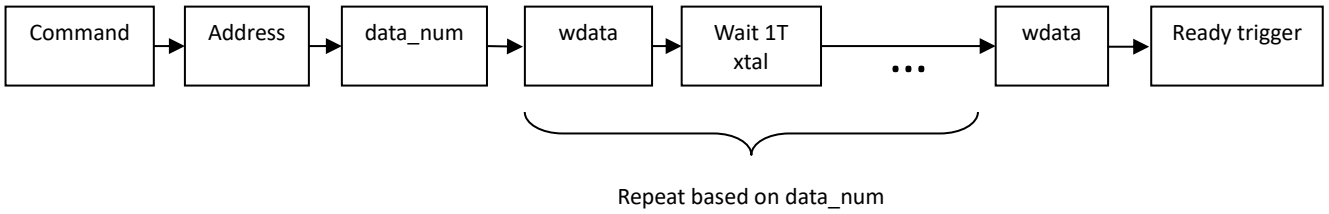


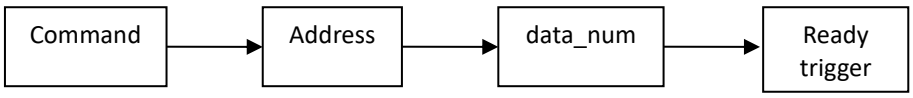
Figure 3-84 DPCD writing flow

dptx_reg_3640_dp_tx_aux[6]: Sink reply complete IRQ

dptx_reg_3640_dp_tx_aux[0]: AUX timeout IRQ
dptx_reg_3624_dp_tx_aux[3:0]: Sink reply command

3.8.5.5.16 DPCD Read

These control registers are the same as DPCD write registers.



dptx_reg_3620_dp_tx_aux[8]: The software reads data pulse.

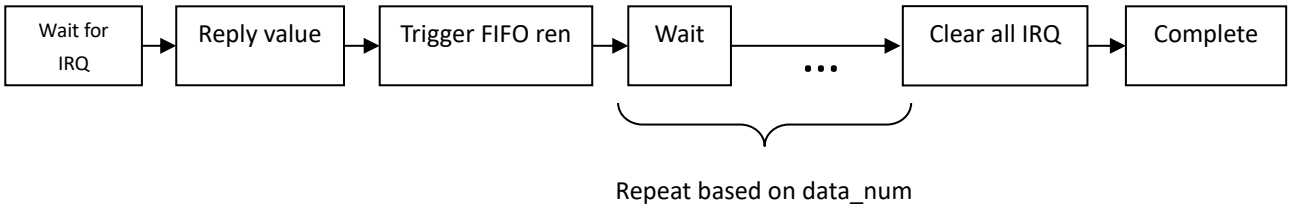


Figure 3-85 DPCD Reading Flow

3.8.5.5.17 I2C Write

The flow is the same as that of DPCD write.

3.8.5.5.18 I2C Write No Length

This flow is different. Since AUX sequence has no length, data_num and wdata can be ignored. And no_length has to be cleared after the communication is completed.
dptx_reg_362C_dp_tx_aux[0]: No length setting

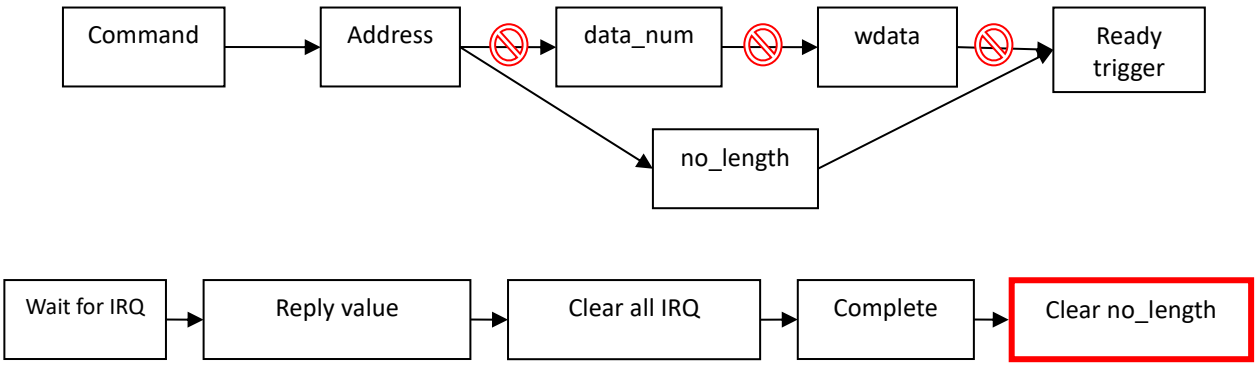


Figure 3-86 I2C Writing Flow

3.8.5.5.19 I2C Read

The flow is the same as that of DPCD read.

3.8.5.5.20 I2C Read No Length

The flow is the same as DPCD write no length, but it is a write command.

3.8.5.5.21 AUX PHY FIFO Status

dptx_reg_368C_dp_tx_aux[0]: RX FIFO done flag
dptx_reg_368C_dp_tx_aux[1]: Clear RX FIFO done flag
dptx_reg_368C_dp_tx_aux[2]: TX FIFO done flag
dptx_reg_368C_dp_tx_aux[3]: Clear TX FIFO done flag

When toggling dptx_reg_3620_dp_tx_aux[8], the software could poll dptx_reg_368C_dp_tx_aux[0]. If this bit is 1'b0, it will read the next data. Before reading the next data, clear the dptx_reg_368C_dp_tx_aux[0] status by dptx_reg_368C_dp_tx_aux[1].

When toggling dptx_reg_3634_dp_tx_aux, the software could poll dptx_reg_368C_dp_tx_aux[2]. If this bit is 1'b0, it will write the next data to this FIFO. Before writing the next data to TX FIFO, clear the dptx_reg_368C_dp_tx_aux[2] status by dptx_reg_368C_dp_tx_aux[3].

3.8.5.5.22 AUX TX Request Pre-charge + Preamble Number

dptx_reg_3630_dp_tx_aux[15:8]: pre-charge + preamble number

3.8.6 High-Definition Multimedia Interface Transmitter (HDMITX)

3.8.6.1 Overview

The HDMITX module encodes video, audio and control data into Transition-Minimized Differential Signaling (TMDS) format for digital transmission based on HDMI Specification 2.0b, and transfers the uncompressed digital data streams to an HDMI- compatible sink device.

3.8.6.2 Features

The HDMITX supports the following video features:

- Deep Color mode: up to 16 bits
- Maximum operating frequency: up to 594 MHz
- Video color space options: RGB444, YCbCr 4:2:2 (ITU 601 and 709), YCbCr 4:4:4 (ITU 601 and 709), YCbCr 4:2:0, and xvYCC
- 3D HDMI function
- SD mode resolutions:
  - 1440 × 480i (pixel repeat 2) @59.94/60 Hz
  - 720 × 480p @ 59.94/60 Hz
  - 1440 × 576i (pixel repeat 2) @50 Hz
  - 720 × 576p @50 Hz
- HD/FHD/UFHD mode resolutions:

- 1280 × 720p @ 59.94/60/50 Hz
- 1920 × 1080i @ 59.94/60/50 Hz
- 1920 × 1080p @ 59.94/60/50 Hz
- 1920 × 1080p @ 23.97/24 Hz
- 1920 × 1080p @ 25 Hz
- 1920 × 1080p @ 29.97/30 Hz
- 3840 × 2160p @ 29.97/30 Hz
- 3840 × 2160p @ 59.94/60/50 Hz

The HDMITX supports the following audio features:

- Single compressed S/PDIF IEC61937 (up to 192 kHz)
- Single LPC S/PDIF IEC60958 (up to 192 kHz and up to 24 bits), 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, 176.4 kHz, 192 kHz
- Multi-channel PCM input (maximum 8 channels)
- DSD audio
- Compressed lossless audio according to HDMI 2.0 (Dolby TrueHD and DTS-HD)

Additionally, the HDMITX supports the following general features:

- HPD line
- Discovery by EDID
- Compatible with DVI 1.0
- HDCP 1.4/HDCP 2.3 function
- Support of dynamic metadata maximum to 2KB
- I^2C-based Display Data Channel (DDC) with clock stretching
- Variable Refresh Rate (VRR)/Auto Low Latency Mode (ALLM)
- CEC channel shared with HDMIRX

3.8.6.3 Block Diagram

Figure 3-87 is the HDMITX block diagram.

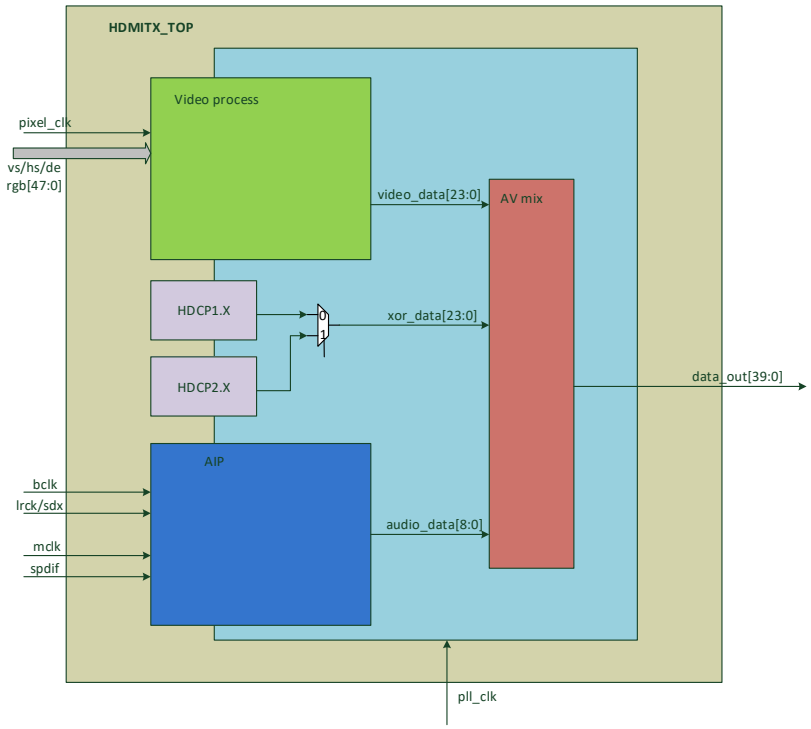


Figure 3-87 HDMITX Block Diagram

3.8.6.4 HDMITX Signal Descriptions

Table 3-72 presents HDMITX signal descriptions.

Table 3-72 HDMITX Signal Descriptions

Signal Name	Type	Description	Ball Location
HDMITX20_CEC	DIO	HDMITX CEC channel (shared with HDMIRX)	AM23
HDMITX20_HTPLG	DI	HDMITX HPD line	AM24
HDMITX20_PWR5V	DO	HDMITX power supply (+5 V)	AL24
HDMITX20_SCL	DIO	HDMITX DDC/I2C clock	AL23
HDMITX20_SDA	DIO	HDMITX DDC/I2C data	AL25
HDMITX21_CH0_M	AO	HDMITX TMDS data lane 0 (negative)	AR32
HDMITX21_CH0_P	AO	HDMITX TMDS data lane 0 (positive)	AR33
HDMITX21_CH1_M	AO	HDMITX TMDS data lane 1 (negative)	AT34
HDMITX21_CH1_P	AO	HDMITX TMDS data lane 1 (positive)	AU34
HDMITX21_CH2_M	AO	HDMITX TMDS data lane 2 (negative)	AR35
HDMITX21_CH2_P	AO	HDMITX TMDS data lane 2 (positive)	AR36
HDMITX21_CLK_M	AO	HDMITX TMDS clock lane (negative)	AT31
HDMITX21_CLK_P	AO	HDMITX TMDS clock lane (positive)	AU31

3.8.6.5 Theory of Operations

Data is transmitted in the same way as described in the specification of HDMI 2.0b.

3.8.6.6 Programming Guide

To start HDMITX function, follow the sequence to program the settings:

1. HDMITX analog clock configuration
2. HDMITX pixel clock configuration
3. Deep color mode selection
4. Audio mode configuration
5. Packet content configuration
6. Optional HDCP configuration

3.8.7 High-Definition Multimedia Interface Receiver (HDMIRX)

3.8.7.1 Overview

The HDMIRX module receives uncompressed digital data streams in TMDS format from an HDMI-compatible source device and decodes the video, audio and control data based on HDMI Specification 2.0b.

3.8.7.2 Features

The HDMIRX supports the following video features:

- Deep Color mode: up to 10 bits
- Maximum operating frequency: up to 594 MHz
- Video color space options: RGB 4:4:4, YCbCr 4:2:2 (ITU BT.601 and BT.709), YCbCr 4:4:4 (ITU BT.601 and BT.709), YCbCr 4:2:0, and xvYCC
- SD mode resolutions:
  - 720 × 480p @ 59.94/60 Hz
  - 720 × 576p @ 50 Hz
- HD/FHD/UFHD mode resolutions:
  - 1280 × 720p @ 59.94/60/50 Hz
  - 1920 × 1080p @ 59.94/60/50 Hz
  - 1920 × 1080p @ 23.97/24 Hz
  - 1920 × 1080p @ 25 Hz
  - 1920 × 1080p @ 29.97/30 Hz
  - 3840 × 2160p @ 29.97/30 Hz
  - 3840 × 2160p @ 59.94/60/50 Hz

The HDMIRX supports the following audio features:

- Single Linear Pulse Code (LPC) S/PDIF IEC60958 (up to 192 kHz and up to 24 bits), 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, 176.4 kHz, 192 kHz
- Multi-channel Pulse Code Modulation (PCM) input (maximum 8 channels)
- Additionally, the HDMIRX supports the following features:
- Internal Extended Display Identification Data (EDID), 512 bytes
- High-bandwidth Digital Content Protection (HDCP) 1.4/HDCP 2.3 key revocation scheme
- Compatibility with Digital Visual Interface (DVI) 1.0

- HDR10+/Dolby dynamic metadata
- Consumer Electronics Control (CEC) channel shared with HDMITX
- Single Hot-Plug Detect (HPD) line

3.8.7.3 Block Diagram

Figure 3-88 is the block diagram of HDMIRX.

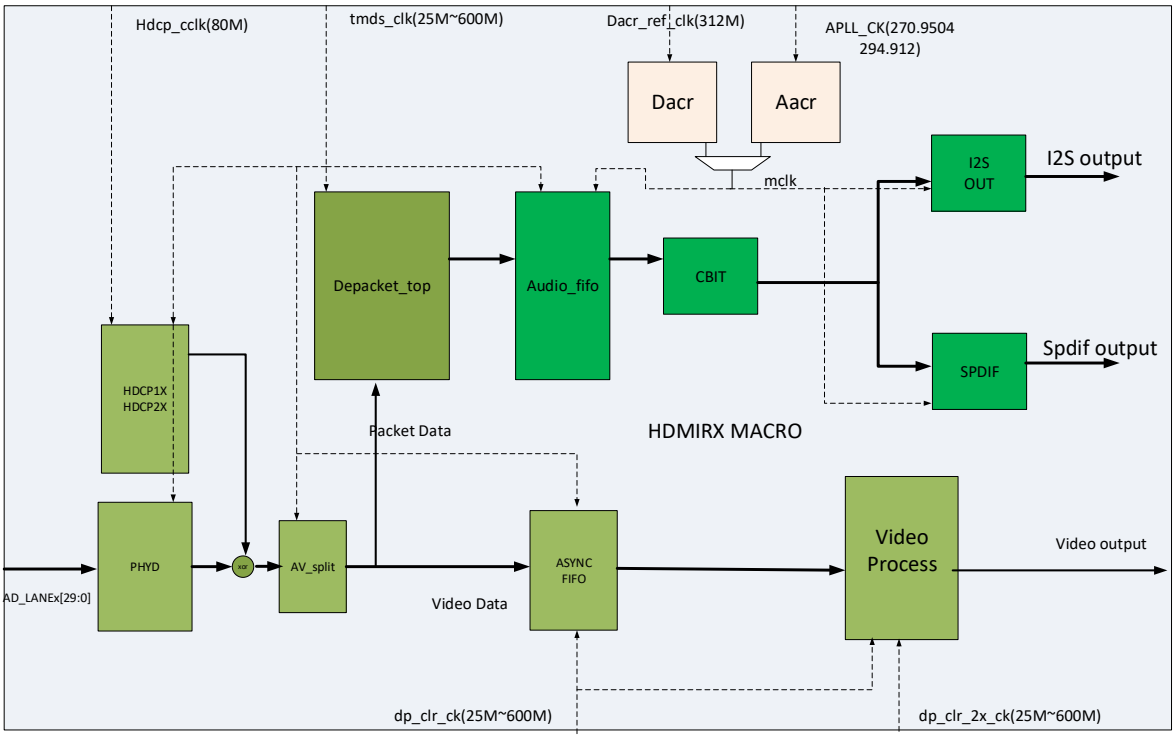


Figure 3-88 HDMIRX Block Diagram

3.8.7.4 HDMIRX Signal Descriptions

Table 3-73 presents HDMIRX signal descriptions.

Table 3-73 HDMIRX Signal Descriptions

Signal Name	Type	Description	Ball Location
HDMIRX20_HTPLG	DO	HDMIRX HPD line	AN25
HDMIRX20_PWR5V	DI	HDMIRX power supply (+5 V)	AN24
HDMIRX20_SCL	DI	HDMIRX DDC/I2C clock	AM25
HDMIRX20_SDA	DIO	HDMIRX DDC/I2C data	AP24
HDMIRX21_CH0_M	AI	HDMIRX TMDS data lane 0 (negative)	AR26
HDMIRX21_CH0_P	AI	HDMIRX TMDS data lane 0 (positive)	AR27
HDMIRX21_CH1_M	AI	HDMIRX TMDS data lane 1 (negative)	AU28
HDMIRX21_CH1_P	AI	HDMIRX TMDS data lane 1 (positive)	AT28
HDMIRX21_CH2_M	AI	HDMIRX TMDS data lane 2 (negative)	AR29
HDMIRX21_CH2_P	AI	HDMIRX TMDS data lane 2 (positive)	AR30

Signal Name	Type	Description	Ball Location
HDMIRX21_CLK_M	AI	HDMIRX TMDS clock lane (negative)	AU25
HDMIRX21_CLK_P	AI	HDMIRX TMDS clock lane (positive)	AT25

3.8.7.5 Theory of Operations

Data is received in the same way as described in the specification of HDMI 2.0b.

3.8.7.6 Programming Guide

To start HDMIRX function, follow the sequence to program the settings:

- HDMIRX analog setting
- HDMIRX PHY setting
- EDID configuration
- HDMIRX MAC setting

3.9 Imaging

The Camera Imaging Subsystem (CAMSYS) is built around a feature-rich Image Signal Processor (ISP) and a deep learning Face Detection (FD) engine. The ISP processes data received either from camera sensors through MIPI CSI-2 interface or system DRAM.

3.9.1 Camera Image Signal Processor (ISP)

The Image Signal Processor (ISP) is responsible for processing the image data generated by the camera sensor of a device. The ISP converts the captured data into a digital format and enhances the image quality before displaying it on the screen or storing it in memory.

3.9.1.1 ISP Features

The ISP supports the following key features:

- 3 × MIPI CSI-2 high-speed camera serial interfaces:
  - Up to 4 × data lanes of MIPI D-PHY 2.5 Gbps per lane
  - Up to 3 × trios of MIPI C-PHY 4.5 Gsps per trio
- Sensor supported format:
  - RAW8/RAW10/RAW12/RAW14
  - YUV422 8-bit (only dump DRAM)
- Single camera capture: up to 48MP at 30fps
- Dual camera capture: up to 16MP + 16MP at 30fps
- ISP image processing capability: 400MP/sec
- Video High Dynamic Range (HDR) with stagger HDR sensor: up to 12MP at 30fps
- Full size image capture for preview
- Image processing functions:

- Auto sensor defect pixel correction
- Lens shading correction
- Edge enhancement
- Video stabilization
- Motion compensated temporal noise reduction for video recording
- Electronic image stabilization
- Multiple frame noise reduction for image capture
- Zero shutter delay image capture
- Preference color adjustment
- AE/AWB/AF statistics collection
- Three-frame stagger HDR fusion
- Color aberration correction
- Anti-blooming correction
- YUV frame buffer compression

3.9.1.2 ISP Block Diagram

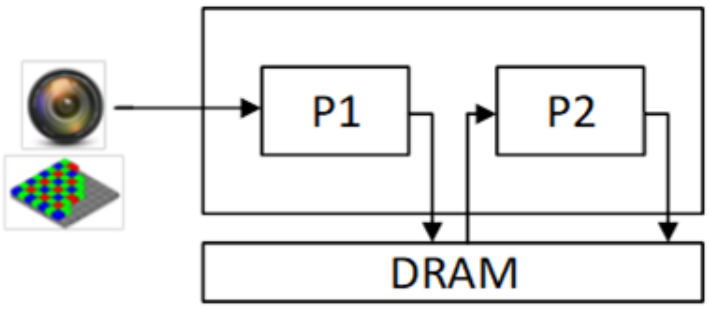


Figure 3-89 Block diagram of ISP

The ISP consists of 2 main engines: Pass 1 (P1) and Pass 2 (P2).

- P1 receives raw data from the camera sensor, executes lens and sensor compensation algorithms and converts the image into YUV format.
- P2 obtains data from P1 through DRAM, and further enhances the image quality, such as noise reduction, preference color adjustment, edge enhancement, etc.

Note:

- The block diagram only represents a typical use case. The path may differ for other scenarios, such as debugging, tuning and engineering modes.

3.9.1.3 Camera Signal Descriptions

Table 3-74 presents camera signal descriptions.

Table 3-74 Camera Signal Descriptions

Signal Name	Type	Description	Ball Location
CMFLASH0	DO	Camera flash strobe 0	AN25

Signal Name	Type	Description	Ball Location
CMFLASH1	DO	Camera flash strobe 1	AN24
CMFLASH2	DO	Camera flash strobe 2	AM25
CMFLASH3	DO	Camera flash strobe 3	AP24
CMMCLK0	DO	Sensor reference clock 0	AL27
CMMCLK1	DO	Sensor reference clock 1	AK27
CMMCLK2	DO	Sensor reference clock 2	AM28
CMMCLK3	DO	Sensor reference clock 3	AL26
CMMCLK4	DO	Sensor reference clock 4	AM26
CMMPDN	DO	Power down to sensor	AM26
CMMRST	DO	Reset control to sensor	AL26
CMVREF0	DO	Camera frame sync 0	AM23
CMVREF1	DO	Camera frame sync 1	AL23
CMVREF2	DO	Camera frame sync 2	AL25
CMVREF3	DO	Camera frame sync 3	H32
CMVREF4	DO	Camera frame sync 4	H34

3.9.2 Face Detection (FD)

3.9.2.1 Overview

The Face Detection (FD) engine uses a convolutional neural network algorithm to detect faces on a source image and output the detected coordinates of the face windows and their confidence values.

3.9.2.2 Features

The FD engine supports the following key features:

- Input image formats:
  - YUV420: 2 planes (Y/UV)
  - YUV422: 2 planes (Y/UV, Y/VU)
  - YUV422: 1 plane (YUYV, YVYU, UYVY, VYUY)
  - Mono 8-bit 1 plane
- YUV to RGB888 format conversion
- Image up/down-scaling
  - Maximum resize width: 640 pixels
- Maximum throughput
  - 800x600@30FPS

3.9.2.3 Block Diagram

Figure 3-90 is the block diagram of FD.

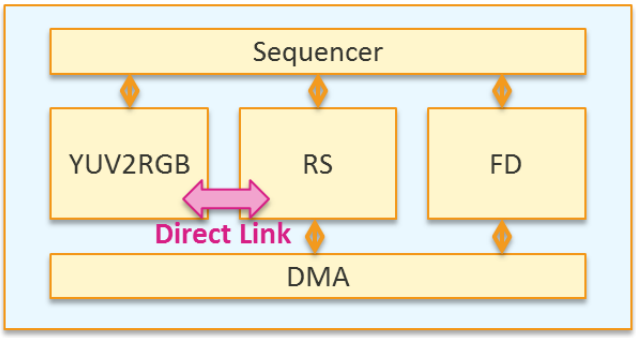


Figure 3-90 Block Diagram of AIE2.0

3.9.2.4 Function Description

The FD is a simple DRAM access engine supporting YUV420 and mono input image and build image pyramid by the up/down-size resizer inside the FD. YUV2RGB is directly linked to RS (resizer) to reduce DRAM bandwidth. After converting YUV420 to RGB888 and building an image pyramid, FD starts several face detection loops and outputs the detected coordinates of the face window, and their confidence values.

3.9.2.5 Programming Guide

Step 1: Program FDVT frame-level registers

To program the FD, the first step is programming frame-level registers by the APB interface. These registers contain the information of configuration DRAM base address, resizer loop number, face detection loop number, etc.

Step 2: Place YUV2RGB, RS, and FD configuration to DRAM

After programming the corresponding configuration DRAM base address, place the YUV2RGB, RS, and FD configuration into DRAM. The sequencer inside the FD goes to the corresponding address to get the configuration during different stages.

Step 3: Trigger FDVT start register

Program the FDVT_START register.

Step 4: Wait for FDVT interrupt

Once the programming of FD is done, an interrupt is generated.

3.9.3 Camera Serial Interface (CSI)

3.9.3.1 Overview

The CSI is based on MIPI Alliance Specification for Camera Serial Interface 2 (MIPI CSI-2) Version 2.1. The CSI provides high-speed serial data transfer between the ISP and external camera image sensors.

The device features two MIPI CSI-2 controllers (CSI0 and CSI1), which are fully compliant with the MIPI CSI-2 specification. The CSI0 controller uses a MIPI D-PHY physical layer, while the CSI1 controller utilizes a combined MIPI D-PHY/C-PHY physical layer. The PHY layer is based on MIPI D-PHY Specification Revision 1.2 and acts as a physical link between the CSI controllers and image sensors.

3.9.3.2 Features

The MIPI CSI-2 implementation in the device provides the following key features:

- Primary CSI-2 interface (CSI0), which can be used in the following configuration:
  - Two 4-data lane interfaces in D-PHY mode with data rate 2.5Gbps
  - One 8-data lane interface in D-PHY mode with data rate 1.44Gbps
- Secondary CSI-2 interface (CSI1), which can be used in one of the following configurations:
  - Two 2-data lane interfaces in D-PHY mode, or
  - One 4-data lane interface in D-PHY mode, or
  - One 3-trio interface in C-PHY mode, or
  - Two 2-trio interfaces in C-PHY mode
  - D-PHY supports 2.5Gbps per lane and C-PHY supports 4.5Gsps per trio
- Pixel formats: RAW8/RAW10/RAW12/RAW14/YUV422 8-bit
- No support for D-PHY escape mode and bus turnaround

3.9.3.3 CSI Signal Descriptions

Table 3-75 presents CSI0 signal descriptions.

Table 3-75 CSI0 Signal Descriptions

Signal Name(1)	Type	Description	Ball Location
		2 × D-PHY 4-lane Mode	
CSI0A_L0N	AIO	CSI0 port 0 data lane 2 (negative)	J36
CSI0A_L0P	AIO	CSI0 port 0 data lane 2 (positive)	J35
CSI0A_L1N	AIO	CSI0 port 0 data lane 0 (negative)	L34
CSI0A_L1P	AIO	CSI0 port 0 data lane 0 (positive)	K35
CSI0A_L2N	AIO	CSI0 port 0 clock lane (negative)	K33
CSI0A_L2P	AIO	CSI0 port 0 clock lane (positive)	K34
CSI0B_L0N	AIO	CSI0 port 0 data lane 1 (negative)	L33
CSI0B_L0P	AIO	CSI0 port 0 data lane 1 (positive)	L32
CSI0B_L1N	AIO	CSI0 port 0 data lane 3 (negative)	M32
CSI0B_L1P	AIO	CSI0 port 0 data lane 3 (positive)	M33
CSI0C_L0N	AIO	CSI0 port 1 data lane 2 (negative)	M35
CSI0C_L0P	AIO	CSI0 port 1 data lane 2 (positive)	N35
CSI0C_L1N	AIO	CSI0 port 1 data lane 0 (negative)	N33
CSI0C_L1P	AIO	CSI0 port 1 data lane 0 (positive)	N34
CSI0C_L2N	AIO	CSI0 port 1 clock lane (negative)	N32
CSI0C_L2P	AIO	CSI0 port 1 clock lane (positive)	N31
CSI0D_L0N	AIO	CSI0 port 1 data lane 1 (negative)	L37
CSI0D_L0P	AIO	CSI0 port 1 data lane 1 (positive)	L36
CSI0D_L1N	AIO	CSI0 port 1 data lane 3 (negative)	M36
CSI0D_L1P	AIO	CSI0 port 1 data lane 3 (positive)	M37

(1) Unused CSI ports could be connected to GND or set in not connected.

Table 3-76 presents CSI1 signal descriptions.

Table 3-76 CSI1 Signal Descriptions

Signal Name[2]	Type	Description			Ball Location
		1 × D-PHY 4-lane Mode	2 × D-PHY 2-lane Mode	1 × C-PHY 3-trio Mode	
CSI1A_L0N_T0B	AIO	CSI1 data lane 2 (negative)	CSI1 port 0 data lane 0 (negative)	CSI1 Trio0 B	R33
CSI1A_L0P_T0A	AIO	CSI1 data lane 2 (positive)	CSI1 port 0 data lane 0 (positive)	CSI1 Trio0 A	R32
CSI1A_L1N_ T1A	AIO	CSI1 data lane 0 (negative)	CSI1 port 0 clock lane (negative)	CSI1 Trio1 A	R35
CSI1A_L1P_T0C	AIO	CSI1 data lane 0 (positive)	CSI1 port 0 clock lane (positive)	CSI1 Trio0 C	R34
CSI1A_L2N_T1C	AIO	CSI1 clock lane (negative)	CSI1 port 0 data lane 1 (negative)	CSI1 Trio1 C	R37
CSI1A_L2P_T1B	AIO	CSI1 clock lane (positive)	CSI1 port 0 data lane 1 (positive)	CSI1 Trio1 B	R36
CSI1B_L0N_T0B	AIO	CSI1 data lane 1 (negative)	CSI1 port 1 data lane 0 (negative)	CSI1 Trio2 B	T37
CSI1B_L0P_T0A	AIO	CSI1 data lane 1 (positive)	CSI1 port 1 data lane 0 (positive)	CSI1 Trio2 A	T36
CSI1B_L1N_T1A	AIO	CSI1 data lane 3 (negative)	CSI1 port 1 clock lane (negative)	-	T34
CSI1B_L1P_T0C	AIO	CSI1 data lane 3 (positive)	CSI1 port 1 clock lane (positive)	CSI1 Trio2 C	T35
CSI1B_L2N_T1C	AIO	-	CSI1 port 1 data lane 1 (negative)	-	T32
CSI1B_L2P_T1B	AIO	-	CSI1 port 1 data lane 1 (positive)	-	T33

(2) Unused CSI ports could be connected to GND or set in not connected.

3.9.3.4 CSI Timing Characteristics

The CSI interface timing characteristics are compliant with MIPI CSI-2 Specification v2.1, MIPI D-PHY Specification v1.2, and MIPI C-PHY Specification v2.0.

Table 3-77 MIPI D-PHY RX Electrical Characteristics

Description	Min	Typ	Max	Unit	Note
High speed date rate	80	-	2500	Mbps	
High speed common point voltage	70	-	330	mV	
High speed differential input high voltage	-	-	40	mV	
High speed differential input low voltage	-40	-	-	mV	
High speed single ended input high voltage	-	-	460	mV	
High speed single ended input low voltage	-40	-	-	mV	
High speed differential input impedance	80	100	125	Ω	
Low power logic 1 input voltage	740	-	-	mV	

Description	Min	Typ	Max	Unit	Note
Low power logic 0 input voltage	-	-	550	mV	
Low power input hysteresis	25	-	-	mV	
Minimum pulse width response	20	-	-	ns	

3.9.3.5 Block Diagram

The block diagram of the CSI modules is shown in Figure 3-91. The DPHY_TOP module is MIPI DPHY v1.2. The seninf controller is used for unpacking the MIPI stream into pixel data format by MIPI CSI2. The seninf_top_mux is used for selecting pixel data generated by the CSI2 controller and transferring them to seninf_mux. The seninf_mux selects one group or test model signal and transfers the pixel data to seninf_cam_mux. Each seninf_cam_mux is associated with ISP, and selects the pixel data from one of the seninf_muxes. The buffer in the seninf_mux serves as a horizontal blanking (temporal) buffer for pixel data, and ISP reads out one or more pixel data from the buffer during the horizontal period, so the ISP clock frequency can be reduced. The Clock System (CKSYS) is used for generating proper clocks to external camera sensor modules (MCLK).

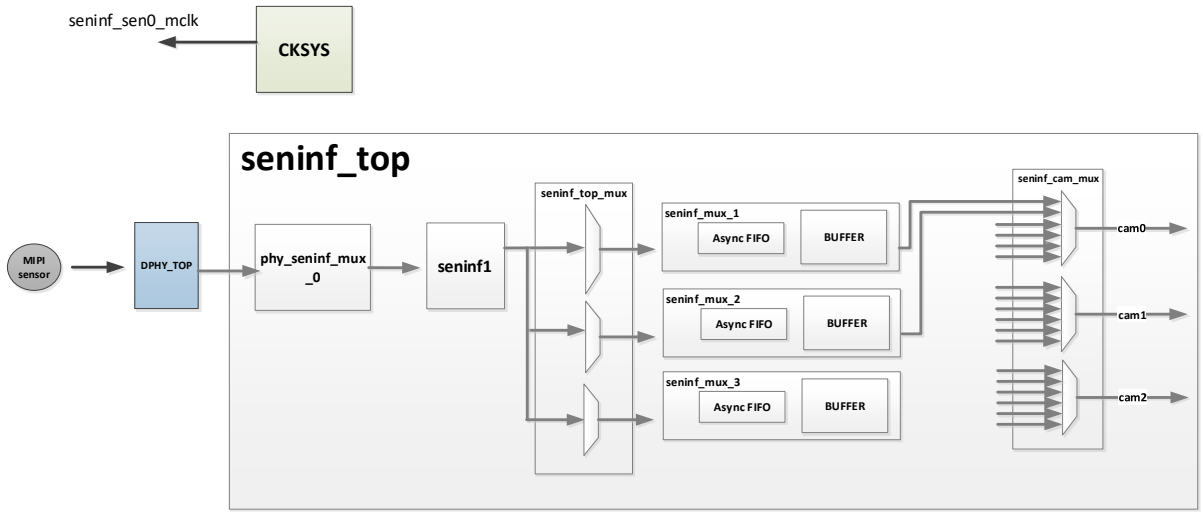


Figure 3-91 CSI csirx_dphy_top and seninf_top Block Diagram (1 Port)

3.9.3.6 Theory of Operations

The block diagram has one DPHY_TOP module, and it can connect with a 4D1C sensor. The seninf1 is responsible for receiving data from sensor and de-packeting the MIPI byte data to pixel data. The seninf_top_mux module can switch different groups to different seninf_muxes. Each seninf_mux has a buffer to store the pixel data generated from seninf.

3.9.3.7 Programming Guide

- CSI programming sequence

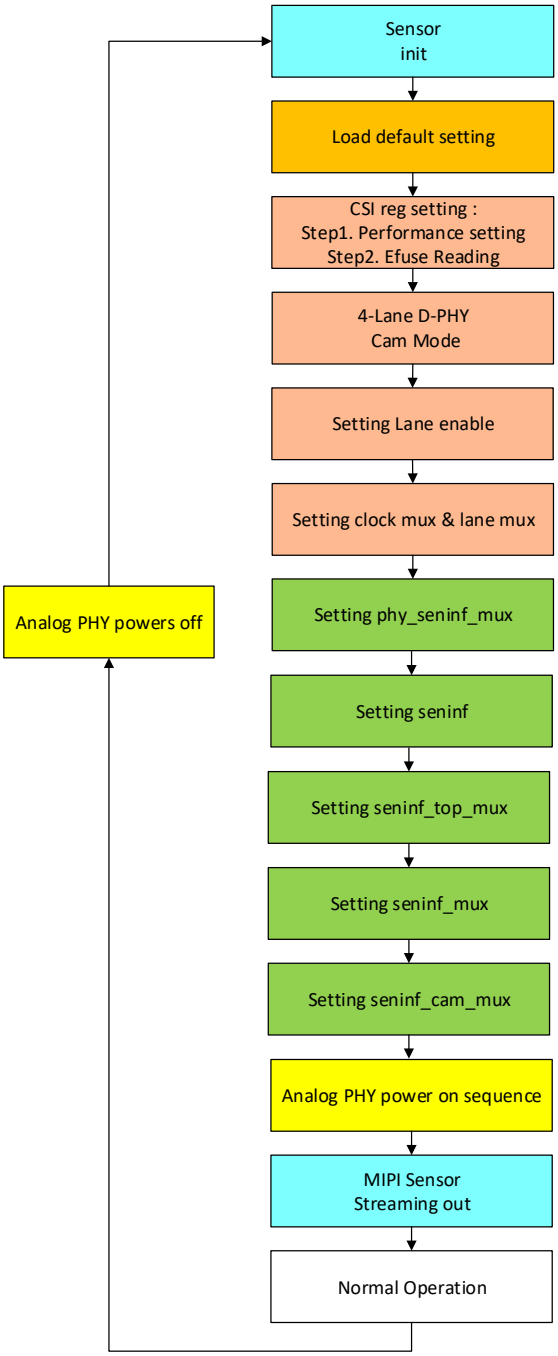


Figure 3-92 CSI Programming Sequence

- CSI basic register setting
  This part takes a basic register setting as an example to show that MIPI sensor data stream can pass through seninf_top to ISP pass1. See the red thread in Figure 3-93.

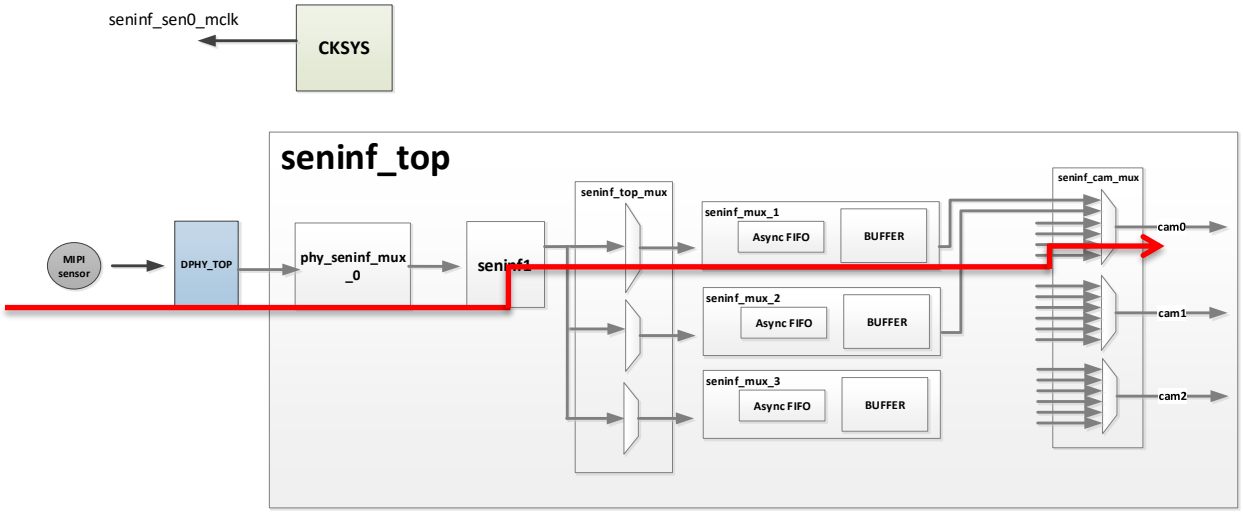


Figure 3-93 Seninf_top Basic Data Path (One Port)

- Load default setting

Table 3-78 Load Default Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Set CSI Register Reset	TOPRGUWDT_SWSYSRST	camsys_rst	1'b1	0x10007018
2	Wait for 1 µs				
3	Release CSI Register Reset	TOPRGUWDT_SWSYSRST	camsys_rst	1'b0	0x10007018
4	Set Seninf_top Clock On	CAMSYS_CG_CON		1'b0	0x14000000
5	Set Seninf_top Reset	CAMSYS_SW_RST	seninf_top_sw_rst	2'b11	0x140000A0
6	Wait for 1 µs				
7	Release Seninf_top Reset	CAMSYS_SW_RST	seninf_top_sw_rst	2'b00	0x140000A0

- Performance setting

Table 3-79 Performance Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	CSI0A performance setting	CSI0A_CDPHY_RX_ANA_1	RG_CSI0A_BG_LPRX_VTL_SEL[2:0]	3'b100	0x11C80004
		CSI0A_CDPHY_RX_ANA_1	RG_CSI0A_BG_LPRX_VTH_SEL[2:0]	3'b100	0x11C80004
		CSI0A_CDPHY_RX_ANA_1	RG_CSI0A_BG_VREF_SEL[3:0]	4'b1000	0x11C80004
2	CSI0B performance setting	CSI0B_CDPHY_RX_ANA_1	RG_CSI0B_BG_LPRX_VTL_SEL[2:0]	3'b100	0x11C81004
		CSI0B_CDPHY_RX_ANA_1	RG_CSI0B_BG_LPRX_VTH_SEL[2:0]	3'b100	0x11C81004
		CSI0B_CDPHY_RX_ANA_1	RG_CSI0B_BG_VREF_SEL[3:0]	4'b1000	0x11C81004

- eFuse reading

  Read eFuse value according to eFuse map, and set the eFuse value to the corresponding registers.

Table 3-80 eFuse Reading

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Read eFuse value	A_CSI_0	rg_csi0a_l0p_t0a_hsrt_code[4:0]	0x11F103A0[4:0]	0x11F103A0
		A_CSI_0	rg_csi0a_l0n_t0b_hsrt_code[4:0]	0x11F103A0[9:5]	0x11F103A0
		A_CSI_0	rg_csi0a_l1p_t0c_hsrt_code[4:0]	0x11F103A0[14:10]	0x11F103A0

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
		A_CSI_0	rg_csi0a_l1n_t1a_hsrt_code[4:0]	0x11F103A0[19:15]	0x11F103A0
		A_CSI_0	rg_csi0a_l2p_t1b_hsrt_code[4:0]	0x11F103A0[24:20]	0x11F103A0
		A_CSI_0	rg_csi0a_l2n_t1c_hsrt_code[4:0]	0x11F103A0[29:25]	0x11F103A0
		A_CSI_1	rg_csi0b_l0p_t0a_hsrt_code[4:0]	0x11F103A4[4:0]	0x11F103A4
		A_CSI_1	rg_csi0b_l0n_t0b_hsrt_code[4:0]	0x11F103A4[9:5]	0x11F103A4
		A_CSI_1	rg_csi0b_l1p_t0c_hsrt_code[4:0]	0x11F103A4[14:10]	0x11F103A4
		A_CSI_1	rg_csi0b_l1n_t1a_hsrt_code[4:0]	0x11F103A4[19:15]	0x11F103A4
2	CSI0A termination control registers setting	CSI0A_CDPHY_RX_ANA_2	RG_CSI0A_L0P_T0A_HSRT_CODE[4:0]	rg_csi0a_l0p_t0a_hsrt_code[4:0]	0x11C80008
		CSI0A_CDPHY_RX_ANA_2	RG_CSI0A_L0N_T0B_HSRT_CODE[4:0]	rg_csi0a_l0n_t0b_hsrt_code[4:0]	0x11C80008
		CSI0A_CDPHY_RX_ANA_3	RG_CSI0A_L1P_T0C_HSRT_CODE[4:0]	rg_csi0a_l1p_t0c_hsrt_code[4:0]	0x11C8000C
		CSI0A_CDPHY_RX_ANA_3	RG_CSI0A_L1N_T1A_HSRT_CODE[4:0]	rg_csi0a_l1n_t1a_hsrt_code[4:0]	0x11C8000C
		CSI0A_CDPHY_RX_ANA_4	RG_CSI0A_L2P_T1B_HSRT_CODE[4:0]	rg_csi0a_l2p_t1b_hsrt_code[4:0]	0x11C80010
		CSI0A_CDPHY_RX_ANA_4	RG_CSI0A_L2N_T1C_HSRT_CODE[4:0]	rg_csi0a_l2n_t1c_hsrt_code[4:0]	0x11C80010
3	CSI0B termination control registers setting	CSI0B_CDPHY_RX_ANA_2	RG_CSI0B_L0P_T0A_HSRT_CODE[4:0]	rg_csi0b_l0p_t0a_hsrt_code[4:0]	0x11C81008
		CSI0B_CDPHY_RX_ANA_2	RG_CSI0B_L0N_T0B_HSRT_CODE[4:0]	rg_csi0b_l0n_t0b_hsrt_code[4:0]	0x11C81008
		CSI0B_CDPHY_RX_ANA_3	RG_CSI0B_L1P_T0C_HSRT_CODE[4:0]	rg_csi0b_l1p_t0c_hsrt_code[4:0]	0x11C8100C
		CSI0B_CDPHY_RX_ANA_3	RG_CSI0B_L1N_T1A_HSRT_CODE[4:0]	rg_csi0b_l1n_t1a_hsrt_code[4:0]	0x11C8100C

- 4-lane DPHY cam mode

Table 3-81 4-Lane DPHY Cam Mode

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Data lane setting	CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_DPHY_L0_CKMODE_EN	1'b0	0x11C80000
		CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_DPHY_L0_CKSEL	1'b1	0x11C80000
		CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_DPHY_L1_CKMODE_EN	1'b0	0x11C80000
		CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_DPHY_L1_CKSEL	1'b1	0x11C80000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_DPHY_L0_CKMODE_EN	1'b0	0x11C81000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_DPHY_L0_CKSEL	1'b1	0x11C81000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_DPHY_L1_CKMODE_EN	1'b0	0x11C81000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_DPHY_L1_CKSEL	1'b1	0x11C81000
2	Clock lane setting	CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_DPHY_L2_CKMODE_EN	1'b1	0x11C80000
		CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_DPHY_L2_CKSEL	1'b1	0x11C80000

- Set lane enable

  CSI 4D1C lane enable top programming outline

Table 3-82 4D1C Lane Enable Top Programming Outline

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Configure clock lane	DPHY_RX_LANE_EN	dphy_rx_lc0_en	1'b1	0x11C82000
2	Set data lane enable	DPHY_RX_LANE_EN	dphy_rx_lc1_en	1'b0	0x11C82000
		DPHY_RX_LANE_EN	dphy_rx_ld0_en	1'b1	0x11C82000
		DPHY_RX_LANE_EN	dphy_rx_ld1_en	1'b1	0x11C82000
		DPHY_RX_LANE_EN	dphy_rx_ld2_en	1'b1	0x11C82000
		DPHY_RX_LANE_EN	dphy_rx_ld3_en	1'b1	0x11C82000
3	Disable software reset	DPHY_RX_LANE_EN	dphy_rx_sw_rst	1'b0	0x11C82000

- Set clock mux and lane mux

CSI 4D1C clock lane and data lane setting top programming outline

Table 3-83 4D1C Clock Lane and Data Lane Setting Top Programming Outline

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	CK_DATA Mux enable	DPHY_RX_LANE_SELECT	dphy_rx_ck_data_mux_en	1'b1	0x11C82004
2	Configure clock lane	DPHY_RX_LANE_SELECT	rg_dphy_rx_lc0_sel	3'h2	0x11C82004
3	Set data lane enable	DPHY_RX_LANE_SELECT	rg_dphy_rx_ld0_sel	3'h1	0x11C82004
		DPHY_RX_LANE_SELECT	rg_dphy_rx_ld1_sel	3'h3	0x11C82004
		DPHY_RX_LANE_SELECT	rg_dphy_rx_ld2_sel	3'h0	0x11C82004
		DPHY_RX_LANE_SELECT	rg_dphy_rx_ld3_sel	3'h4	0x11C82004
4	Configure settle time	DPHY_RX_CLOCK_LANE0_HS_PARAMETER	rg_dphy_rx_lc0_hs_settle_parameter	8'h28	0x11C82010
		DPHY_RX_DATA_LANE0_HS_PARAMETER	rg_cdphy_rx_ld0_trio0_hs_settle_parameter	8'h28	0x11C82020
		DPHY_RX_DATA_LANE1_HS_PARAMETER	rg_cdphy_rx_ld1_trio1_hs_settle_parameter	8'h28	0x11C82024
		DPHY_RX_DATA_LANE2_HS_PARAMETER	rg_cdphy_rx_ld2_trio2_hs_settle_parameter	8'h28	0x11C82028
		DPHY_RX_DATA_LANE3_HS_PARAMETER	rg_cdphy_rx_ld3_trio3_hs_settle_parameter	8'h28	0x11C8202C

- phy_seninf_mux_setting

Table 3-84 Phy_seninf_mux_0 Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Enable seninf_top D-PHY mux 0	SENINF_TOP_PHY_CTRL_CSI0	phy_seninf_mux0_dphy_en	1'b1	0x14040060

- seninf setting

Table 3-85 Seninf1 Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Enable CSI2	SENINF_CSI2_CTRL	rg_seninf_csi2_en	1'b1	0x14040210
2	Enable seninf	SENINF_CTRL	seninf_en	1'b1	0x14040200
3	Enable CSI2 lane numbers	SENINF_CSI2_EN	csi2_lane0_en csi2_lane1_en csi2_lane2_en csi2_lane3_en	4'b1111	0x14040A00
4	Assert CSI2 software reset	SENINF_CSI2_CTRL	seninf_csi2_sw_rst	1'b1	0x14040210
5	De-assert CSI2 software reset			1'b0	

- seninf_top_mux setting

Table 3-86 Seninf_top_mux Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Seninf top mux 1 input source selection	SENINF_TOP_MUX_CTRL_0	rg_seninf_mux1_src_sel	4'b0000	0x14040010

- seninf_mux_1 setting

Table 3-87 Seninf_mux_1 Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Disable seninf mux 1	SENINF_MUX_CTRL_0	seninf_mux_en	1'b0	0x14040D00
2	Assert seninf mux 1 software reset		seninf_mux_sw_rst	1'b1	

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
3	Assert seninf mux 1 IRQ software reset		seninf_mux_irq_sw_rst	1'b1	
4	De-assert seninf mux 1 IRQ software reset		seninf_mux_irq_sw_rst	1'b0	
5	De-assert seninf mux 1 software reset		seninf_mux_sw_rst	1'b0	
6	Select seninf mux 1 input source	SENINF_MUX_CTRL_1	rg_seninf_mux_src_sel	4'b1000	0x14040D04
7	Enable seninf mux 1	SENINF_MUX_CTRL_0	seninf_mux_en	1'b1	0x14040D00

- seninf_cam_mux setting

Table 3-88 Seninf_cam_mux Setting

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	Enable seninf cam mux 0	SENINF_CAM_MUX_EN	seninf_cam_mux0_en	1'b1	0x14040410
2	Disable seninf cam mux 1		seninf_cam_mux1_en	1'b0	
3	Disable seninf cam mux 2		seninf_cam_mux2_en	1'b0	

- Analog PHY power on sequence

Table 3-89 Analog Power-On Sequence

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	BG Core Enable	CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_BG_CORE_EN	1'b1	0x11C80000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_BG_CORE_EN	1'b1	0x11C81000
2	Wait > 30 µs				
3	BG LPF Enable	CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_BG_LPF_EN	1'b1	0x11C80000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_BG_LPF_EN	1'b1	0x11C81000

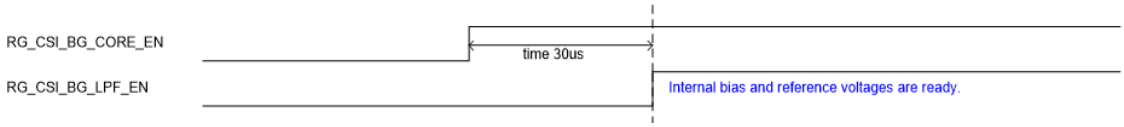


Figure 3-94 Power-On Sequence Timing

- Analog PHY powers off

Table 3-90 Analog Power-Off

Step	Description	Register Name	Register Bit Field	Bit Field Value	Address
1	BG Core and LPF Disable	CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_BG_CORE_EN	1'b0	0x11C80000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_BG_CORE_EN	1'b0	0x11C81000
		CSI0A_CDPHY_RX_ANA_0	RG_CSI0A_BG_LPF_EN	1'b0	0x11C80000
		CSI0B_CDPHY_RX_ANA_0	RG_CSI0B_BG_LPF_EN	1'b0	0x11C81000

3.9.3.8 External Sensor Clock Option

This section describes the master clock (mclk) option for the image sensor. In most cases, the clock system of the external image sensor module requires a reference clock source. The mclk can be obtained from different clock sources that are not integrated in the CSI controller. The cksys provides a variety of mclk options from internal clock system. The supported clock can be 6/12/13/24/26/48/52 MHz.

Table 3-91 CAM_CLK Pad Configuration

Pin Name	Aux Func.0	Aux Func.1	Aux Func.2	Aux Func.3
CAM_CLK	B:GPIO 183	O:CAM_CLK		

In GPIO Aux function 1, this pin is connected to the f_fcamtg_ck clock mux.

- CLK_CFG_3 register 0x10000040[7] controls whether f_fcamtg_ck is disabled or not.
- CLK_CFG_3 register 0x10000040[2:0] controls the MUX to select clock. Therefore, the different clock sources can be configured as follows:

Table 3-92 CAMTG Function Clock Selection

Address	Register Name	Description
0x10000040	CLK_CFG_3	Function Clock Selection Register
Bit(s) Field	**Name**	**Description**
2: 0	clk_camtg_sel	Selects f_fcamtg_ck clock MUX
		0: 26M (Xtal 26M) 1: 24M (UNIVPLL_192M/8) 2: 52M (UNIVPLL/8) 3: 48M (UNIVPLL_192M/4) 4: 26M (UNIVPLL/16) 5: 13M (Xtal 26M/2) 6: 12M (UNIVPLL_192M/16) 7: 6M (UNIVPLL_192M/32)

3.10 Video

The device has two video accelerators—Video Encoder (VENC) and Video Decoder (VDEC).

3.10.1 Video Encoder (VENC)

3.10.1.1 Overview

The VENC accelerator supports main stream H.264 and HEVC video encoding. It is capable of encoding 4K video at 60 fps superior video quality. The VENC supports various encoding methods that satisfy basic requirements of easy control by software. The VENC brings astonishing high quality and low memory bandwidth requirements, with advanced encoding technology. The accelerator also considers the usage of portable devices and provides several power saving capabilities.

3.10.1.2 Features

The VENC has the following main features:

- Uses DRAM as an input, output, and working buffer
- Reads input frame buffers, executes video encoding and writes encoded bit stream to the output buffer
- Support of YUV420 two-plane scan line (NV12/NV21) and YUV420 three-plane scan line (YV12/I420) color spaces

- Support RGB and ARGB input formats.

Table 3-93 presents the supported video formats and their capabilities.

Table 3-93 VENC Supported Formats

Format	Feature	Details
H.264 Encoding	Profile	High (10-bit)
	Level	L5.2
	Speed	4K @ 60 fps (100 Mbps)
HEVC Encoding	Profile	Main (10-bit)
	Level	L5.1
	Speed	4K @ 60 fps (100 Mbps)

3.10.1.3 Block Diagram

Figure 3-95 is the brief IP architecture and local on-chip-bus architecture. The interface for controlling it consists of ARM APB and MediaTek proprietary SMI (Smart Multimedia Interface) bus. It reports a hardware event through an interrupt or software polling. In addition, it adopts several SMI ports and one APB port. The video encoder core includes the following modules: DMA, ME, MC, TQ, DB and EC. The input to video encoder is image data. After the encoding process, the bitstream is sent to DRAM by system bus.

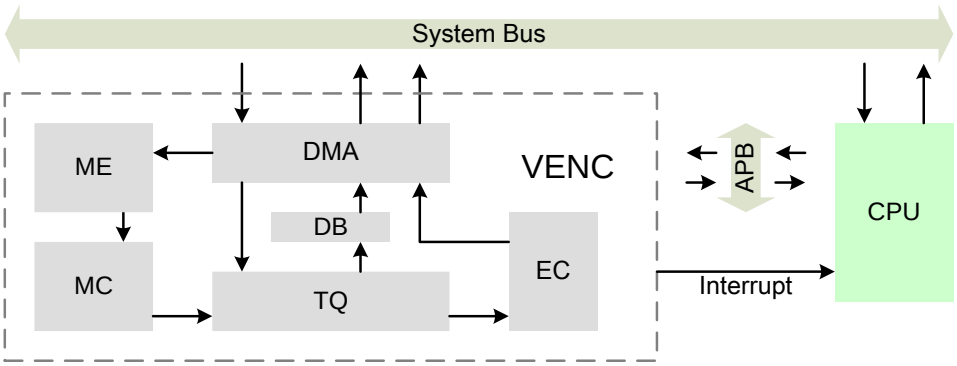


Figure 3-95 Block Diagram of VENC

3.10.1.4 Function Description

The video encoder is configured by software through the APB interface. As the register is configured, the sequencer sends the corresponding control signals to trigger sub-modules.

- **DMA** acquires and stores back the image data and bitstream from and to memory according to the configured address.
- **ME** conducts motion estimation to decide motion vector for later encoding.
- **MC** conducts motion compensation to give predicted pixel values.
- **TQ** conducts transform and quantization operation and writes reconstructed pixels to DB and quantized transformed coefficient to EC.
- **DB** conducts de-blocking operation and allows DMA to store back the processed frame as the next frame's reference frame.

- **EC** conducts entropy encoding, and the coding can be variable length code, context based arithmetic code, or context based variable length code. The encoded bitstream is written to memory by DMA.

3.10.2 Video Decoder (VDEC)

3.10.2.1 Overview

The Video Decoder (VDEC) accelerator is designed to provide multi-standard video decoding, relieving the Central Processing Unit (CPU) load and providing high performance video. It takes a compressed video bitstream as input, performs the decoding process and then sends the reconstructed video to the display.

3.10.2.2 Features

- HEVC
  - Main profile 4K2K @ 90fps / 160 Mbps, 8 bits
  - Main 10 profile 4K2K @ 90fps / 160 Mbps, 10 bits
- HEIF
  - Main profile maximum resolution 16383 × 16383, 8 bits
  - Main10 profile maximum resolution 16383 × 16383, 10 bits
- VP9
  - Profile 0 4K2K @ 90 fps / 120 Mbps, 8 bits
  - Profile 2 4K2K @ 90 fps / 120 Mbps, 10 bits
- AV1
  - Main profile 0 4K2K @ 90 fps / 120 Mbps, 8/10 bits
- AVC
  - Constrained Baseline 4K2K @ 90fps / 160 Mbps, 8 bits
  - Main /High profile 4K2K @ 90fps / 160 Mbps, 8 bits
  - High 10 profile 4K2K @ 90fps / 160 Mbps, 10 bits
- MPEG-4
  - Simple Profile 1080p @ 60 fps / 60 Mbps, 8 bits
  - Advanced Simple Profile 1080p @ 60 fps / 60 Mbps, 8 bits
- MPEG-2
  - Main profile 1080p @ 60 fps / 60 Mbps, 8 bits
- VP8
  - 1080p @ 60 fps / 40 Mbps, 8 bits
- H.263
  - Baseline profile 1080p @ 60 fps / 60 Mbps

3.11 Audio

3.11.1 Overview

The MediaTek audio system allows data exchange among the AP and external components. The following interfaces are available.

- Two I2S/Time Division Multiplexing (TDM) outputs
- Two I2S/TDM inputs
- One High-Definition Multimedia Interface (HDMI™) TX
- One DisplayPort (DP) TX output
- One HDMIRX audio input
- One S/PDIF input
- One S/PDIF output
- One PCM interface with Sampling Rate Converter (SRC)
- One slave 8-channel I2S input (AUDIO IN)
- Four Pulse Density Modulation (PDM) interfaces for Digital Microphone Interface Controller (DMIC)
- One proprietary audio interface for the MediaTek PMIC codec

3.11.2 Features

- PMIC audio CODEC playing
  - Supports 8/11.025/12/16/22.05/24/32/44.1/48/96/192 kHz sampling rate playing.
- PMIC audio CODEC recording
  - Supports 8/16/32/48/96/192 kHz sampling rate recording.
- One master I^2S output (I2SO1):
  - 24-channel I^2S output. Sampling rates from 8 kHz to 384 kHz, up to 32 bits
  - 16-channel TDM output. Sampling rates from 8 kHz to 48 kHz, up to 32 bits
  - 6-channel Direct Stream Digital (DSD) with clock rate of 2.8 MHz or 2-channel DSD with clock rates of 5.6 MHz or 11.2 MHz
- One master or slave I^2S output (I2SO2):
  - 8-channel I^2S output. Sampling rates from 8 kHz to 384 kHz, up to 32 bits
  - 24-channel TDM output with 16 kHz sampling rate or 16-channel @ 48 kHz
- One master or slave I^2S input (I2SIN):
  - 8-channel I^2S input. Sampling rates from 8 kHz to 384 kHz and resolution up to 32 bits
  - 8-channel TDM input. Sampling rates from 8 kHz to 48 kHz, up to 32 bits, or 16-channel direct path to memory
- One master or slave TDM input (TDMIN):
  - 24-channel TDM output with 16 kHz sampling rate and bit resolution up to 32 bits or 16-channel @ 48 kHz
  - 2-channel I^2S input. Sampling rates from 8 kHz to 384 kHz and resolution up to 32 bits
- One master 8-channel High-Definition Multimedia Interface (HDMI) audio output (HDMITX):
  - Sampling rates from 8 kHz to 192 kHz with resolution of up to 24 bits
  - 6-channel DSD Out with clock rate of 2.8 MHz or 2-channel DSD Out with clock rates of 5.6 MHz or 11.2 MHz
- One master 8-channel DisplayPort TX output with sampling rates from 8 kHz to 192 kHz with bit resolution up to 24 bits.
- HDMIRX audio input with Direct Stream Digital (DSD) support
- One S/PDIF input. Sampling rates include 32, 44.1, 48, 88.2, 96, 176.4, and 192 kHz
- One S/PDIF output with 32, 44.1, 48, 88.2, 96, and 192 kHz sampling rates
- One master or slave PCM interface with Sampling Rate Converter (SRC). Supported sampling rates: 8, 16, 32, 44.1, and 48 kHz with bit resolution up to 24 bits

- One slave 8-channel I^2S input (AUDIO IN) with sampling rate from 8 kHz to 192 kHz with bit resolution up to 24 bits
- Four Pulse Density Modulation (PDM) interfaces for up to 4 stereo Digital Microphones (DMICs). Support of one-wire and two-wire modes with 8, 16, 32, 48, 96 and 192 kHz PCM sampling rates with bit resolution of 24 bits
- A proprietary audio interface for PMIC CODEC
  - 2-channel DAC. Supports up to 192 kHz sampling rate
  - 3-channel ADC. Supports up to 192 kHz sampling rate
- Hardware gain function and general ASRC to enhance the audio quality and flexibility of mix engine.
- The flexible mix engine system for data exchange between the interfaces without CPU intervention

3.11.3 Block Diagram

Figure 3-96 illustrates the flexibility of the mix engine between audio interfaces.

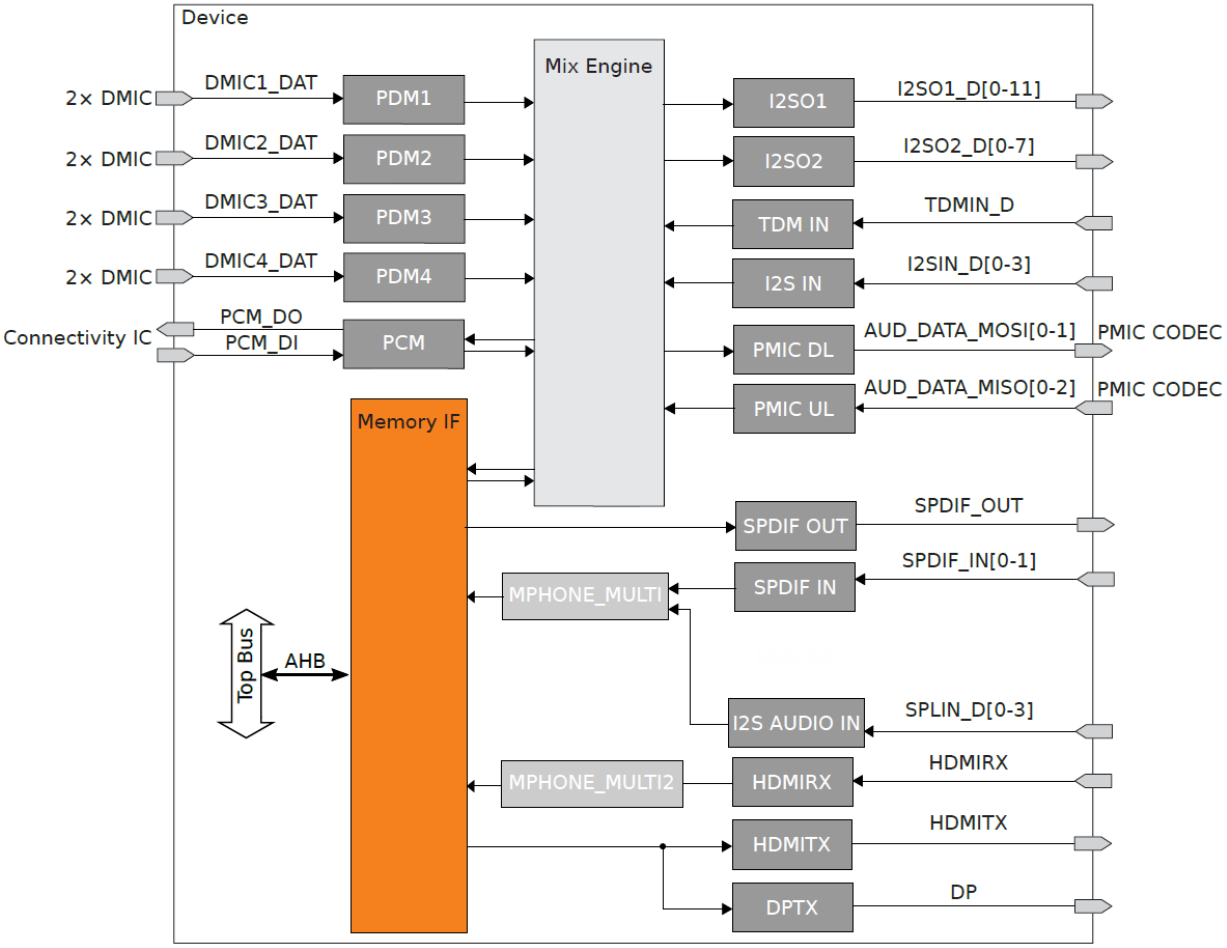


Figure 3-96 Audio Interfaces Block Diagram

3.11.4 Function Description

3.11.4.1 Inter-IC Sound (I2S)

3.11.4.1.1 I2S Signal Descriptions

Table 3-94 presents I2S signal descriptions.

Table 3-94 I2S Signal Descriptions

Signal Name	Type	Description	Ball Location
I2SIN			
I2SIN_BCK	DIO	I2SIN serial bit clock	AK20, AL10
I2SIN_D0	DI	I2SIN serial data input 0	AP20, AM8
I2SIN_D1	DI	I2SIN serial data input 1	AT12, AT10
I2SIN_D2	DI	I2SIN serial data input 2	AT14, AR8
I2SIN_D3	DI	I2SIN serial data input 3	AP8, AT9
I2SIN_MCK	DIO	I2SIN master clock	AL20, AT11
I2SIN_WS	DIO	I2SIN word select (left/right audio channel)	AR20, AT9
I2SO1			
I2SO1_BCK	DO	I2SO1 serial bit clock	AK19
I2SO1_D0	DO	I2SO1 serial data output 0	AR18
I2SO1_D1	DO	I2SO1 serial data output 1	AP18
I2SO1_D2	DO	I2SO1 serial data output 2	AN18
I2SO1_D3	DO	I2SO1 serial data output 3	AM18
I2SO1_D4	DO	I2SO1 serial data output 4	H32
I2SO1_D5	DO	I2SO1 serial data output 5	H34
I2SO1_D6	DO	I2SO1 serial data output 6	E37
I2SO1_D7	DO	I2SO1 serial data output 7	F37
I2SO1_D8	DO	I2SO1 serial data output 8	AM8
I2SO1_D9	DO	I2SO1 serial data output 9	AT10
I2SO1_D10	DO	I2SO1 serial data output 10	AR8
I2SO1_D11	DO	I2SO1 serial data output 11	AP8
I2SO1_MCK	DO	I2SO1 master clock	AN19
I2SO1_WS	DO	I2SO1 word select (left/right audio channel)	AT19
I2SO2			
I2SO2_BCK	DIO	I2SO2 serial bit clock	AK18, AP17
I2SO2_D0	DO	I2SO2 serial data output 0	AT20, AT6
I2SO2_D1	DO	I2SO2 serial data output 1	AN12, AT7
I2SO2_D2	DO	I2SO2 serial data output 2	AU14, AT8
I2SO2_D3	DO	I2SO2 serial data output 3	AT13, AU8
I2SO2_MCK	DO	I2SO2 master clock	AL18
I2SO2_WS	DIO	I2SO2 word select (left/right audio channel)	AU20, AL17
SPLIN			
SPLIN_BCK	DI	SPLIN serial bit clock	AR20
SPLIN_D0	DI	SPLIN data 0	AP20
SPLIN_D1	DI	SPLIN data 1	AP18

Signal Name	Type	Description	Ball Location
SPLIN_D2	DI	SPLIN data 2	AN18
SPLIN_D3	DI	SPLIN data 3	AM18
SPLIN_LRCK	DI	SPLIN word select	AK20
SPLIN_MCK	DI	SPLIN master clock	AL20

3.11.4.1.2 I2S Timing Characteristics

Table 3-95, Figure 3-97 and Figure 3-98 present timing characteristics for the I2S modules in the device.

Table 3-95 I2S Timing Characteristics

No.	Parameter	Description	Min	Typ	Max	Unit
-	f_S	Sampling frequency	8	-	384	kHz
IIS01	f_{c_MCK}	Cycle time, MCK (master clock)	-	-	24.576	MHz
-	f_{OP_BCK}	Operation frequency, BCK	32 × f_S	-	64 × f_S	MHz
IIS03	t_{c_BCK}	Cycle time, BCK	81	-	3906	ns
IIS04	$t_{w\_BCK\_H}$	Pulse duration, BCK high	-	0.5	-	$1/t_{c\_BCK}$
IIS05	$t_{w\_BCK\_L}$	Pulse duration, BCK low	-	0.5	-	$1/t_{c\_BCK}$
-	t_{LRCK}	LRCK period	32	-	64	$1/t_{c\_BCK}$
IIS06	t_{v_LRCK}	BCK negative edge to LRCK valid	-	-	0.2	$1/t_{c\_BCK}$
IIS07	t_{v_DO}	BCK negative edge to DO valid	-	-	0.2	$1/t_{c\_BCK}$
IIS08	t_{su}	Setup time, DI	0.2	-	-	$1/t_{c\_BCK}$
IIS09	t_{su}	Setup time, DI (slave mode)	0.2	-	-	$1/t_{c\_BCK}$
IIS10	t_h	Hold time, DI	0.2	-	-	$1/t_{c\_BCK}$
IIS11	t_h	Hold time, DI (slave mode)	0.2	-	-	$1/t_{c\_BCK}$

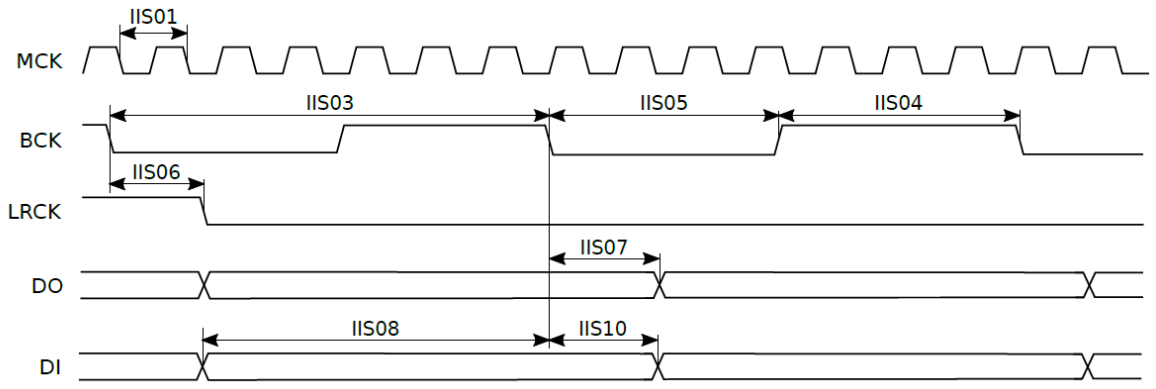


Figure 3-97 I2S Master Mode Timing Diagram

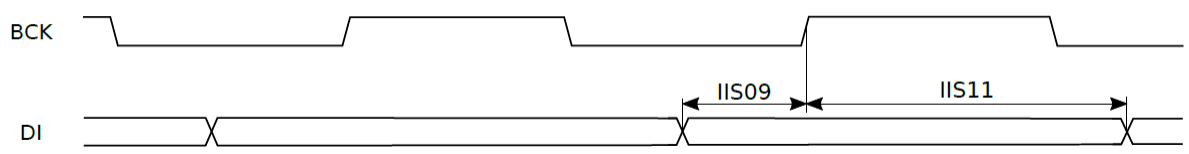


Figure 3-98 I2S Slave Mode Timing Diagram

3.11.4.2 Pulse Code Modulation (PCM)

3.11.4.2.1 PCM Signal Descriptions

Table 3-96 presents PCM signal descriptions.

Table 3-96 PCM Signal Descriptions

Signal Name	Type	Description	Ball Location
PCM_CLK	DIO	PCM clock	AP16
PCM_DI	DI	PCM data input	AN16
PCM_DO	DO	PCM data output	AR16
PCM_SYNC	DIO	PCM synchronization	AM16

3.11.4.2.2 PCM Timing Characteristics

Table 3-97, Figure 3-99 and Figure 3-100 present timing characteristics for the PCM interfaces in the device.

Table 3-97 PCM Timing Characteristics

No.	Parameter	Description	Min	Typ	Max	Unit
-	f_S	Sampling frequency	8	-	48	kHz
PCM1	f_{CLK}	Serial clock frequency	0.256	-	3.072	MHz
-	t_{SYNC}	Sync period	32	-	64	$1/f_{CLK}$
PCM2	$t_{w\_CLK\_H}$	Pulse duration, CLK high	-	0.5	-	$1/f_{CLK}$
PCM3	$t_{w\_CLK\_L}$	Pulse duration, CLK low	-	0.5	-	$1/f_{CLK}$
PCM4	$t_{d\_CLK\_SYNC}$	Delay time, output CLK low to SYNC valid	-	-	0.2	ns
PCM5	$t_{d\_CLK\_TX}$	Delay time, output CLK low to TX valid	-	-	0.2	ns
PCM6	t_{su}	Setup time, RX master mode	0.2	-	-	ns
PCM7	t_h	Hold time, RX master mode	0.2	-	-	ns
PCM8	t_{su}	Setup time, RX slave mode	0.2	-	-	ns
PCM9	t_h	Hold time, RX slave mode	0.2	-	-	ns

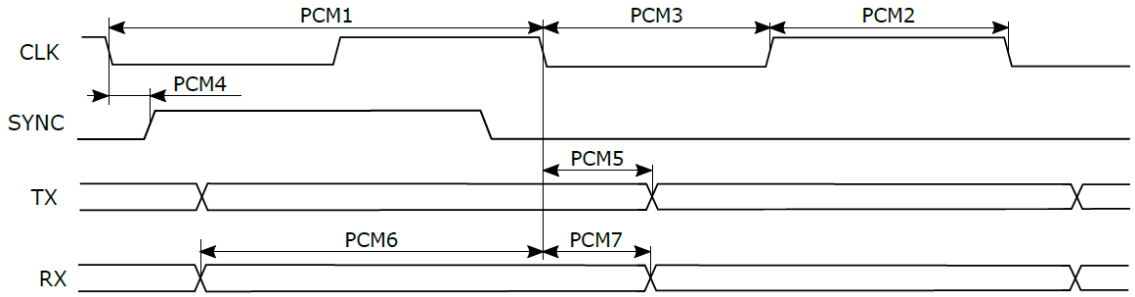


Figure 3-99 PCM Master Mode Timing Diagram

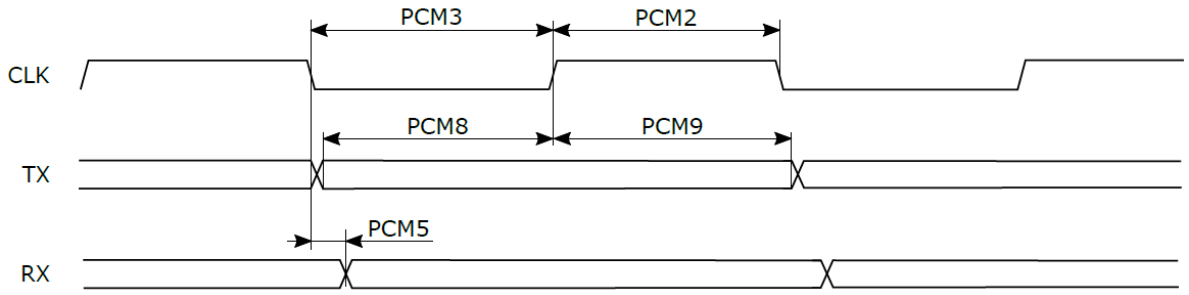


Figure 3-100 PCM Slave Mode Timing Diagram

3.11.4.3 Time Division Multiplexed (TDM) Interface

3.11.4.3.1 TDM Signal Descriptions

Table 3-98 presents TDM signal descriptions.

Table 3-98 TDM Signal Descriptions

Signal Name	Type	Description	Ball Location
TDMIN_BCK	DIO	TDM clock	G33, G36
TDMIN_DI	DI	TDM receive data input	G34, F34
TDMIN_LRCK	DIO	TDM word select (left/right audio channel)	J31, F36
TDMIN_MCK	DIO	TDM receive master clock	H31, F35

3.11.4.3.2 TDM Timing Characteristics

Table 3-99 and Figure 3-101 present timing characteristics for the TDM interfaces in the device. present timing characteristics for the TDM interfaces in the device.

Table 3-99 TDM Timing Characteristics

No.	Parameter	Description	Min	Typ	Max	Unit
-	f_S	Sampling frequency	8	-	192	kHz
TDM1	f_{MCK}	Master clock frequency	0.768	-	49.152	MHz
TDM2	f_{BCK}	Serial clock frequency	0.256	-	49.152	MHz
TDM3	$t_{w\_BCK\_H}$	Pulse duration, BCK high	-	0.5	-	$1/f_{BCK}$
TDM4	$t_{w\_BCK\_L}$	Pulse duration, BCK low	-	0.5	-	$1/f_{BCK}$
TDM5	$t_{d\_BCK\_WS}$	Delay time, output BCK low to WS valid	-	-	0.2	ns
TDM6	$t_{d\_BCK\_SDOUT}$	Delay time, output BCK low to SDOUT valid	-	-	0.2	ns
TDM7	t_{su_DI}	Setup time, DI	0.2	-	-	ns
TDM8	t_{h_DI}	Hold time, DI	0.2	-	-	ns

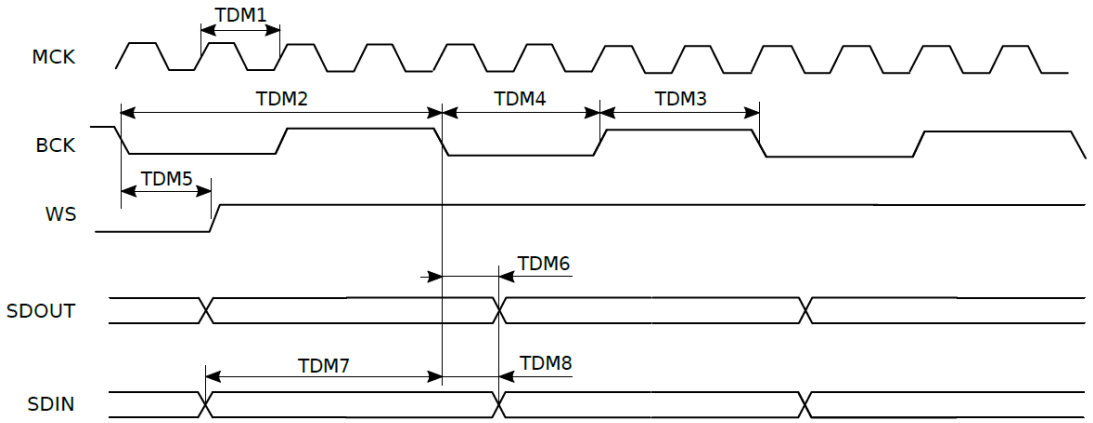


Figure 3-101 TDM Master Mode Timing Diagram

3.11.4.4 Pulse Density Modulation (PDM)

3.11.4.4.1 PDM Timing Characteristics

Table 3-100 and Figure 3-102 present timing characteristics for the PDM interface in the device.

Table 3-100 PDM Timing Characteristics

No.	Parameter	Description	Min	Typ	Max	Unit
-	f_{CLK}	Operating frequency, PDM CLK	0.40625	-	3.25	MHz
PDM2	$t_{W\_CLK\_H}$	Pulse duration, CLK high	-	0.5	-	$1/f_{CLK}$
PDM3	$t_{W\_CLK\_L}$	Pulse duration, CLK low	-	0.5	-	$1/f_{CLK}$
PDM4	t_{SU_DAT}	Setup time, DAT	0.2	-	-	$1/f_{CLK}$
PDM5	t_{H_DAT}	Hold time, DAT	0.2	-	-	$1/f_{CLK}$

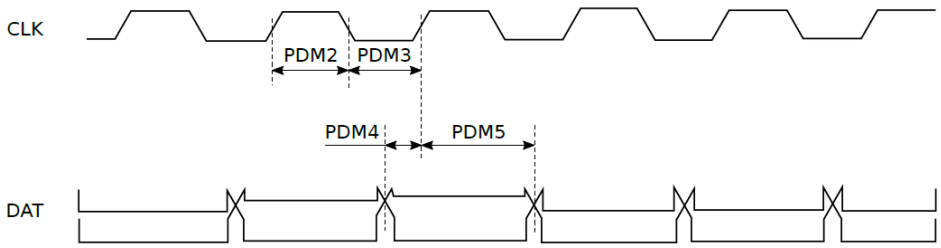


Figure 3-102 PDM Timing Diagram

3.11.4.4.2 Digital Microphone (DMIC)

3.11.4.4.3 DMIC Signal Descriptions

Table 3-101 presents DMIC signal descriptions.

Table 3-101 DMIC Signal Descriptions

Signal Name	Type	Description	Ball Location
DMIC1_CLK	DO	DMIC1 clock	AK27, AP17
DMIC1_DAT	DI	DMIC1 data in one-wire mode or data left	AM28, AL17
DMIC1_DAT_R	DI	DMIC1 data right	AL24
DMIC2_CLK	DO	DMIC2 clock	J31, AU18
DMIC2_DAT	DI	DMIC2 data in one-wire mode or data left	G33, AT18
DMIC2_DAT_R	DI	DMIC2 data right	AL21
DMIC3_CLK	DO	DMIC3 clock	H32, F34, AP22
DMIC3_DAT	DI	DMIC3 data in one-wire mode or data left	H34, F35, AN23
DMIC3_DAT_R	DI	DMIC3 data right	E37
DMIC4_CLK	DO	DMIC4 clock	H31, G36, AP23
DMIC4_DAT	DI	DMIC4 data in one-wire mode or data left	G34, F36, AN22
DMIC4_DAT_R	DI	DMIC4 data right	F37

3.11.4.5 Digital Interface (SPDIF)

3.11.4.5.1 SPDIF Signal Descriptions

Table 3-102 presents SPDIF signal descriptions.

Table 3-102 SPDIF Signal Descriptions

Signal Name	Type	Description	Ball Location
SPDIF_IN0	DI	SPDIF input 0	H33, E35
SPDIF_IN1	DI	SPDIF input 1	G35, E36
SPDIF_OUT	DO	SPDIF output	D36, AL26

3.11.4.6 Power Management Integrated Circuit (PMIC)

3.11.4.6.1 PMIC Audio Interface Signal Descriptions

Table 3-103 presents PMIC audio interface signal descriptions.

Table 3-103 PMIC Audio Interface Signal Descriptions

Signal Name	Type	Description	Ball Location
AUD_CLK_MOSI(1)	DO	PMIC CODEC clock master output	AL16
AUD_DAT_MISO0	DI	PMIC CODEC data master input 0	AK15
AUD_DAT_MISO1	DI	PMIC CODEC data master input 1	AU17
AUD_DAT_MISO2	DI	PMIC CODEC data master input 2	AT17
AUD_DAT_MOSI0(1)	DO	PMIC CODEC data master output 0	AT16
AUD_DAT_MOSI1(1)	DO	PMIC CODEC data master output 1	AN15
AUD_SYNC_MOSI(1)	DO	PMIC CODEC sync master output	AK16

(1) These pins should be left unconnected when unused.

3.12 Connectivity

3.12.1 Inter-Integrated Circuit (I2C) and Improved I2C (I3C)

3.12.1.1 Overview

The I3C/I2C controller is a bi-directional, two-wire serial interface that utilizes Serial Clock Line (SCL) and Serial Data Line (SDA) signals. These signals can be driven by either the master or the slave in both I2C and I3C, with the exception that SCL is only driven by the master in I3C. This generic controller supports the master role and conforms to the I3C/I2C specification.

3.12.1.2 Features

- Start, repeated start and stop conditions generation
- Bus detection
- Acknowledge bit generation and detection
- 7-bit/10-bit addressing
- Clock stretching
- Active drive/wired-and I/O configuration
- I2C four operating speeds:
  - I2C Standard mode (SM) with speed up to 100 kbit/s
  - I2C Fast mode (FM) with speed up to 400 kbit/s
  - I2C Fast mode Plus (FM+) with speed up to 1 Mbit/s
  - I2C High Speed (HS) mode with speed up to 3.4 Mbit/s
- Adjustable clock speed for SM/FM/FM+/HS mode operation
- I3C 12.5 MHz Single Data Rate (SDR) mode:
  - Dynamic address assignment
  - Private Write/Read transfer
  - Broadcast Common Command Code (CCC)
  - Directed CCC
- Multi-users
- FIFO mode and DMA mode
- Multiple transfer formats:
  - Multi-write per transfer
  - Multi-read per transfer
  - Multi-transfer per transaction
  - Combined format transfer with length change capability
  - Combined format transfer with direction change capability
  - Multi-transfer with repeated start condition

3.12.1.3 I2C/I3C Signal Descriptions

Table 3-104 presents I2C/I3C signal descriptions.

Table 3-104 I2C/I3C Signal Descriptions

Signal Name	Type	Description	Ball Location
I3C0(1)			
SCL0	DIO	I3C0 serial clock	D37
SDA0	DIO	I3C0 serial data	D36
I3C1(1)			
SCL1	DIO	I3C1 serial clock	E35
SDA1	DIO	I3C1 serial data	E36
I2C2(1)			
SCL2	DIO	I2C2 serial clock	H33, F37
SDA2	DIO	I2C2 serial data	G35, E37
I2C3(1)			
SCL3	DIO	I2C3 serial clock	F34
SDA3	DIO	I2C3 serial data	F35
I2C4(1)			
SCL4	DIO	I2C4 serial clock	G36, AM28
SDA4	DIO	I2C4 serial data	F36, AK27
I2C5			
SCL5	DIO	I2C5 serial clock	AM25, AN20
SDA5	DIO	I2C5 serial data	AP24, AM20
I2C6			
SCL6	DIO	I2C6 serial clock	AL28, AM26, AL23
SDA6	DIO	I2C6 serial data	AM29, AL26, AL25
I2C7			
SCL7	DIO	I2C7 serial clock	AN25, AL23
SDA7	DIO	I2C7 serial data	AN24, AL25

(1) These modules have hardware default internal 10 kΩ pull-up changed by the software initialization to 1 kΩ and external pull-up resistors are not required.

3.12.1.4 Block Diagram

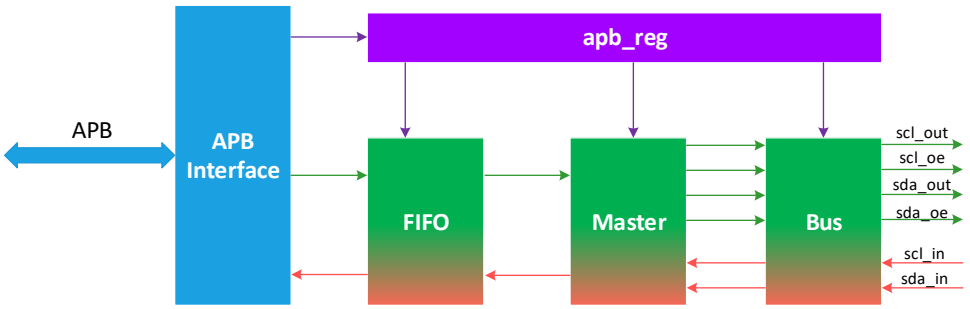


Figure 3-103 Block Diagram of I3C/I2C

The I3C/I2C master controller is composed of several sub-modules, including:

- **APB interface**: Facilitates communication with the host side.
- **FIFO**: 16-byte storage depth
- **apb_reg**: APB register, configuration for other sub-modules.

- **Master**: Read and write control, state machine, timing control, etc.
- **Bus**: Monitors the bus behavior, data output.

The data flow direction differs in the following two scenarios:

- When the I3C/I2C master writes data to the slave, the data flow direction is:
  APB interface ➔ FIFO ➔ Master ➔ Bus ➔ Slave
- When the I3C/I2C master reads data from the slave, the data flow direction is:
  Slave ➔ Bus ➔ Master ➔ FIFO ➔ APB interface

3.12.1.5 Function Description

Figure 3-104 shows the wording conventions that are used subsequently.

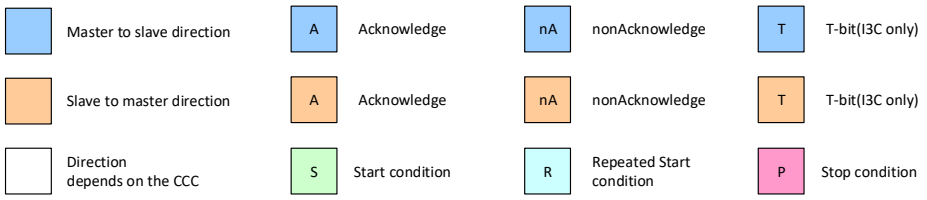


Figure 3-104 I2C/I3C Function Description Wording Conventions

3.12.1.5.1 Start, Repeated Start and Stop Conditions

In I2C and I3C protocols, transactions are beginning by a Start condition (S) and terminated by a Stop condition (P).

Transaction	Description
Start condition	When bus is free, a high to low transition on the SDA line while SCL is high.
Repeat Start condition	When bus is busy, a high to low transition on the SDA line while SCL is high.
Stop condition	When bus is busy, a low to high transition on the SDA line while SCL is high.

3.12.1.5.2 Bus Detection

The I2C master is equipped to detect the bus status. Prior to controlling the bus to transfer data, the master initiates a detection process. If the bus is busy, the master waits until a stop condition is detected.

3.12.1.5.3 I2C Clock Stretching

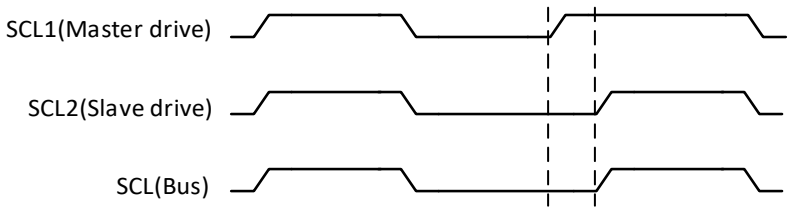


Figure 3-105 I2C Clock Stretching

In certain scenarios, the master may need to communicate with the slave, but the slave may not be ready to receive or send data. This can occur when the internal FIFO of the slave is full or empty. In such cases, after the master pulls SCL low, the slave will actively and continuously pull down the SCL, thus enabling clock stretching.

3.12.1.5.4 I2C Standard Mode (SM), Fast Mode (FM), and Fast Mode Plus (FM+)

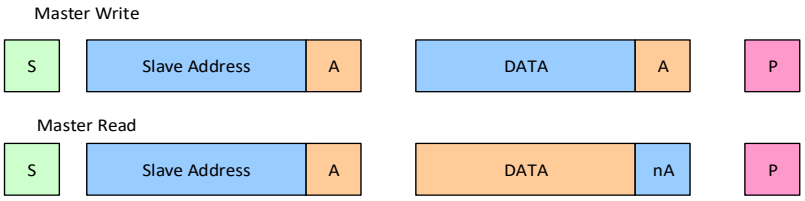


Figure 3-106 I2C Frame Format in SM/FM/FM+

Figure 3-106 illustrates the basic transfer format utilized by the SM/FM/FM+ of I2C. A more complex transfer format is described in Section 3.12.1.8.

Initially, the master sends a start condition. Following this, the master sends the 7-bit static address of the I2C slave device with which it intends to communicate. Once the slave responds to the addressing, the master sends or receives the data. Upon completion of the data transfer, the master sends a stop condition and the bus returns to the free state.

3.12.1.5.5 I2C High-Speed (HS) Mode

Figure 3-107 I2C Write Frame Format in HS Mode

Figure 3-107 depicts the differences between the frame formats employed by the I2C HS mode and I2C SM/FM/FM+. Notably, the I2C HS mode introduces a fixed master code (00001XXX) that broadcasts to all slaves supporting the HS mode, allowing them to transition to high speed. The master then sends a repeat start condition signal followed by the same frame format as in SM/FM/FM+.

3.12.1.5.6 I3C Single Data Rate (SDR) Mode

The I3C master supports the I3C Spec v1.0 SDR mode, including:

- Dynamic Address Assignment (DAA) (3.12.1.5.6.1)
- Private write/read (3.12.1.5.6.1)
- Broadcast CCC (3.12.1.5.6.2)
- Direct CCC (3.12.1.5.6.3)

3.12.1.5.6.1 Dynamic Address Assignment (DAA)

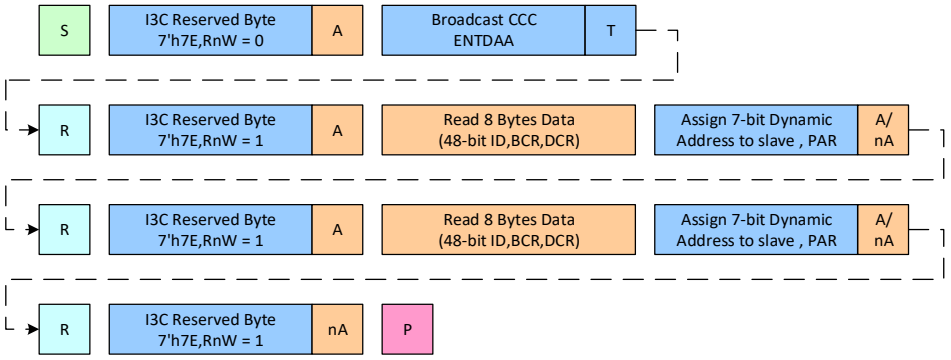


Figure 3-108 I3C Dynamic Address Assignment (DAA)

The procedure of DAA is employed to assign dynamic addresses to I3C slaves, as all I3C slaves must have a dynamic address to complete data transfer.

Private Write/Read

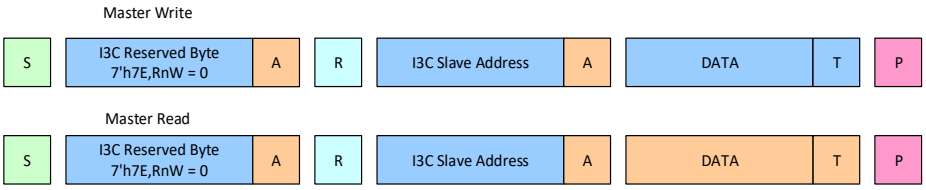


Figure 3-109 I3C SDR Private Write/Read

SDR private write/read is a basic communication mode similar to the I2C HS mode with the frame format shown in Figure 3-109, where 7E is the I3C reserved address to indicate that this message is an I3C message.

3.12.1.5.6.2 Broadcast CCC

Figure 3-110 I3C SDR Broadcast CCC

The I3C master controller supports broadcast CCC (command codes 0x00 to 0x7F) to write to all I3C slaves on the I3C bus.

3.12.1.5.6.3 Direct CCC

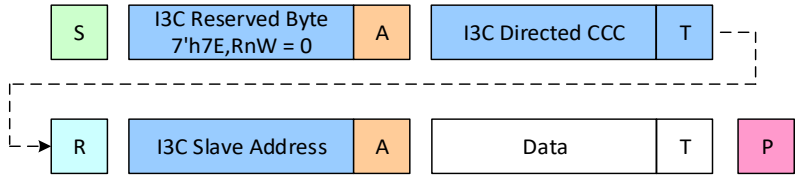


Figure 3-111 I3C SDR direct CCC

The master controller supports direct CCC (command codes 0x80 to 0xFE) to the specific I3C slave on the I3C Bus.

3.12.1.6 Interrupts

Table 3-105 I3C/I2C Interrupt Control Bits and Interrupt Factors

Interrupt Type	Interrupt Enable Bit	Interrupt Identification	Interrupt Factors
Transmit	MAS_TRANSAC_COMP	TRANSAC_COMP	After transmission is completed, the interrupt will be generated.
Communication Error	MAS_ACKERR	ACKERR	When the slave sends no ack, the interrupt will be generated.
Communication Error	MAS_ HS_NACKERR	HS_NACKERR	This status is asserted if HS master code NACK (Negative Ack.) error detection is enabled.
Enhancement Feature	MAS_ARB_LOST	ARB_LOST	When the I2C controller loses arbitration, the interrupt will be generated.
Enhancement Feature	MAS_RS_MULTIPLE	RS_MULTIPLE	In multiple read/write with rs_stop transfer mode, this status is asserted when a transfer has completed successfully.
Enhancement Feature	MAS_TIMEOUT	TIMEOUT	This status is asserted if the I2C bus time-out occurs.
Enhancement Feature	MAS_ DMA_ERR	DMA_ERR	This status is asserted if the DMA hand-shake error has occurred.

3.12.1.7 Theory of Operations

3.12.1.7.1 Multi-User

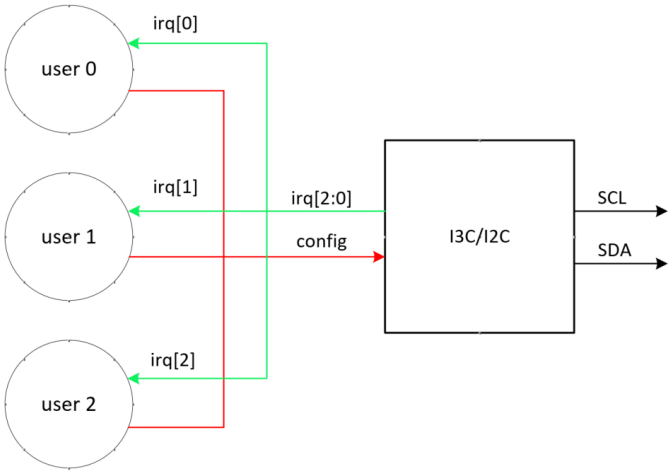


Figure 3-112 Multi-user of I3C/I2C

The I3C/I2C master supports the multi-user function, enabling multiple hosts (including CCU, AP, SCP, etc.) to poll to control the I3C/I2C master, as illustrated in Figure 3-112. In case of using AP/SCP multi-user, it is advised to use the FIFO mode on the non-XPU side. Moreover, if the AP multi-user is employed, the IRQ of the AP's I3C/I2C channel 2 can only be sent to the AP and CCU, and not to the SCP.

3.12.1.7.2 FIFO Mode

The I3C/I2C master controller offers a FIFO mode. Along with the slave address register, the controller integrates a 16-byte deep FIFO. This feature enables the AP to prepare up to 16 bytes of data for a write transfer or read up to 16 bytes of data for a read transfer.

3.12.1.7.3 DMA Mode

The AP_DMA facilitates communication between the I2C master controller and the system. To enable efficient data transfer, configuration of the DMA settings is necessary in advance. These settings include the target memory address and the TX/RX transfer direction and transfer length, in addition to the slave address register.

3.12.1.8 Supported Transfer Formats

The I3C/I2C master controller is designed to be generic to support a wide range of devices that may utilize different combinations of transfer formats. The supported formats and examples are described in the following sections.

Wording conventions:

Word	Description
Transfer	Anything encapsulated within a START and STOP or repeated START.
Transfer length	Number of bytes within the transfer (TRANSFER_LEN (I2Cn Base address + 0x14)/TRANSFER_LEN_AUX (I2Cn Base address + 0x6C)).
Transaction	The top unit. Everything combined equals 1 transaction.
Transaction length	Number of transfers to be conducted (TRANSAC_LEN (I2Cn Base address + 0x18)).

3.12.1.8.1 Single-Byte Access

The I2C master writes 1-byte data to the I2C slave or reads 1-byte data from it. The I3C master is compatible with this transfer format.

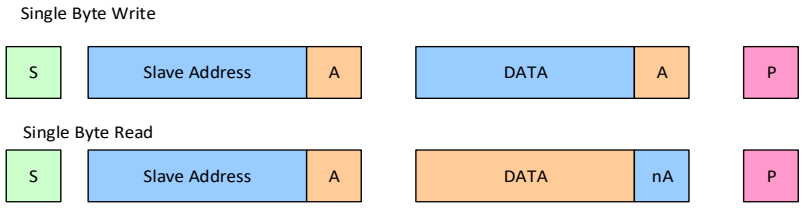


Figure 3-113 I2C/I3C Single-Byte Access

3.12.1.8.2 Multi-Byte Access

The I2C master writes N-byte data to the I2C slave or reads N-byte data from it. The I3C master is compatible with this transfer format.

- DIR_CHANGE(I2Cn Base address + 0x10)[4] = 0
- TRANSAC_LEN(I2Cn Base address + 0x18) = 1
- TRANSFER_LEN (I2Cn Base address + 0x14) = N (N ≥ 1)

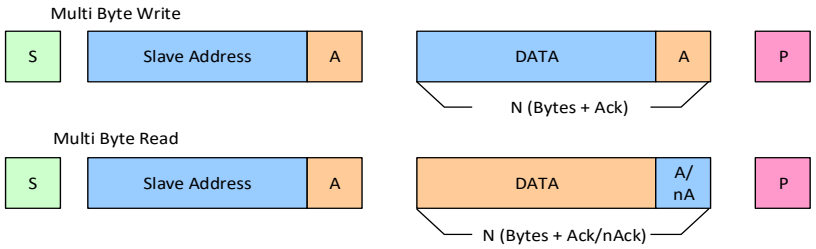


Figure 3-114 I2C/I3C Single-Byte Access

3.12.1.8.3 Multi-Byte Transfer + Multi-Transfer (Same Direction)

The I2C master performs X transfers and each transfer contains N-byte write/read operation. The transfers are separated by a STOP condition and wait time. The I3C master is compatible with this transfer format.

- DIR_CHANGE(I2Cn Base address + 0x10)[4] = 0
- TRANSAC_LEN(I2Cn Base address + 0x18) = X (X ≥ 1)
- TRANSFER_LEN(I2Cn Base address + 0x14) = N (N ≥ 1)
- RS_STOP(I2Cn Base address + 0x10)[1] = 0
- DELAY_LENGTH(I2Cn Base address + 0x1C) = Wait Time

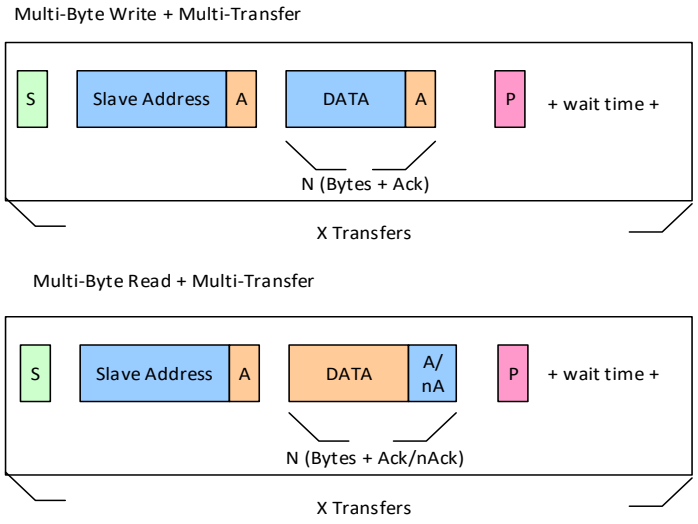


Figure 3-115 I2C/I3C Multi-Byte Transfer + Multi-Transfer (Same Direction)

3.12.1.8.4 Multi-Byte Transfer + Multi-Transfer with Repeated START (Same Direction)

The I2C master performs X transfers and each transfer contains N-byte write/read operation. The transfers are separated by a repeated START condition. The I3C master is compatible with this transfer format.

- DIR_CHANGE(I2Cn Base address + 0x10)[4] = 0
- TRANSAC_LEN(I2Cn Base address + 0x18) = X (X ≥ 1)
- TRANSFER_LEN(I2Cn Base address + 0x14) = N (N ≥ 1)

- RS_STOP(I2Cn Base address + 0x10)[1] = 1

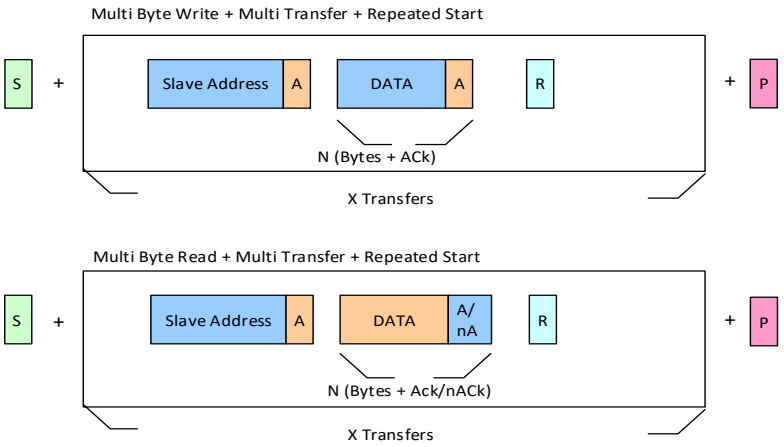


Figure 3-116 I2C/I3C Multi-Byte Transfer + Multi-Transfer with RS (Same Direction)

3.12.1.8.5 Combined Write/Read with Repeated START (Direction Changed)

The I2C master first writes N-byte data to the I2C slave. After a repeated START condition, the I2C master reads M-byte data from the I2C slave. The I3C master is compatible with this transfer format. Note that only the “read after write sequence” is supported; the “write after read sequence” is not supported.

- DIR_CHANGE(I2Cn Base address + 0x10)[4] = 1
- TRANSAC_LEN(I2Cn Base address + 0x18) = 2
- TRANSFER_LEN(I2Cn Base address + 0x14) = N (N ≥ 1)
- TRANSFER_LEN_AUX(I2Cn Base address + 0x6C) = M (M ≥ 1)
- RS_STOP(I2Cn Base address + 0x10)[1] = 1

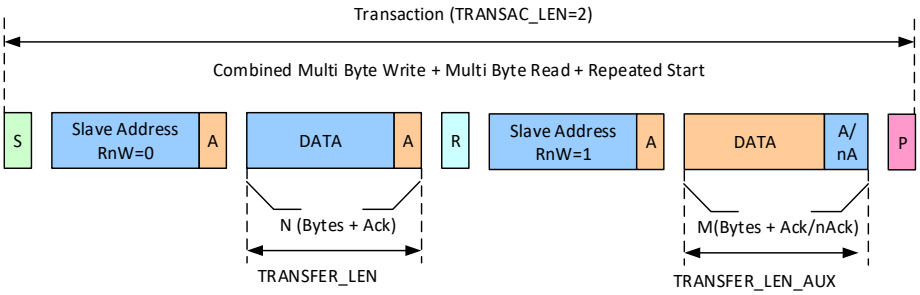


Figure 3-117 Combined Write/Read with Repeated START (Direction Changed)

3.12.1.8.6 Transfer Length Change After First Transfer Completion (No Direction Changed)

Under the condition of changing the transfer length without changing the transfer direction in the write/read mode, the I2C master first writes N-byte data to the I2C slave. After a repeated START condition, the I2C master writes/reads M-byte data to the I2C slave. The I3C master is compatible with this transfer format.

- DIR_CHANGE(I2Cn Base address + 0x10)[4] = 0
- TRANSFER_LEN_CHANGE(I2Cn Base address + 0x10)[6] = 1

- TRANSAC_LEN(I2Cn Base address + 0x18) = 2
- TRANSFER_LEN(I2Cn Base address + 0x14) = N (N ≥ 1)
- TRANSFER_LEN_AUX(I2Cn Base address + 0x6C) = M (M ≥ 1)
- RS_STOP(I2Cn Base address + 0x10)[1] = 1

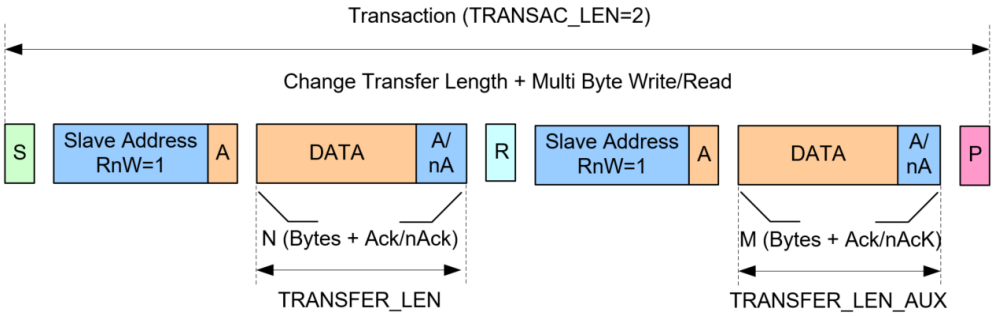


Figure 3-118 Change Transfer Length in Write/Read Mode (No Direction Change)

3.12.1.8.7 Repeated START for Multiple Transfers

When the I2C master performs X transfers, between each two transfers, it is in the pause state and issues an interrupt. At this time, software can change the length, slave address and direction of the next transfer and then restart the transfer. The I3C master is compatible with this transfer format. Note that this function is only supported in the FIFO mode.

- TRANSAC_LEN(I2Cn Base address + 0x18) = X (X ≥ 1)
- TRANSFER_LEN(I2Cn Base address + 0x14) = N (N ≥ 1)
- RS_STOP(I2Cn Base address + 0x10)[1] = 1
- RS_STOP_MULTIPLE_CONFIG(I2Cn Base address + 0x24)[15] = 1
- RS_STOP_MULTIPLE_CONFIG(I2Cn Base address + 0x24)[15] = 0 before last transfer

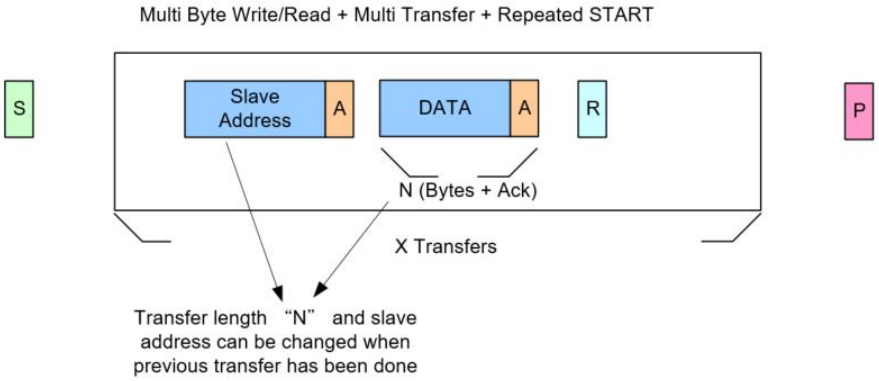


Figure 3-119 Repeated START Multiple Transfers (Write/Read Mode)

3.12.1.9 AC Timing

Table 3-106, Figure 3-120, Table 3-107 and Figure 3-121 present timing characteristics for the I2C interfaces in in SS/FS/FS+ modes.

Table 3-106 I2C AC Timing Parameters for Standard, Fast, and Fast-mode Plus

Symbol	Parameter	Standard-mode		Fast-mode		Fast-mode Plus		Unit	Remark
		Min	Max	Min	Max	Min	Max		
fSCL	Serial Clock Line (SCL) clock frequency	0	100	0	400	0	1000	kHz	-
tHD;STA	Hold time (repeated) START condition	4.0	-	0.6	-	0.26	-	µs	-
tLOW	LOW period of the SCL clock	4.7	-	1.3	-	0.5	-	µs	-
tHIGH	HIGH period of the SCL clock	4.0	-	0.6	-	0.26	-	µs	-
tSU;STA	Set-up time for a repeated START condition	4.7	-	0.6	-	0.26	-	µs	-
tHD;DAT	Data hold time	5.0	-	0	-	0	-	µs	I2C-bus devices
tSU;DAT	Data set-up time	250	-	100	-	50	-	ns	-
tr	Rise time of both SDA and SCL signals	-	1000	20	300	-	120	ns	-
tf	Fall time of both SDA and SCL signals	-	300	20x (VDD/5.5V)	300	20x (VDD/5.5V)	120	ns	VDD is I2C IO voltage
tSU;STO	Set-up time for STOP condition	4.0	-	0.6	-	0.26	-	ns	-
tVD;DAT	Data valid time	-	3.45	-	0.9	-	0.45	µs	-
tVD;ACK	Data valid acknowledge time	-	3.45	-	0.9	-	0.45	µs	-

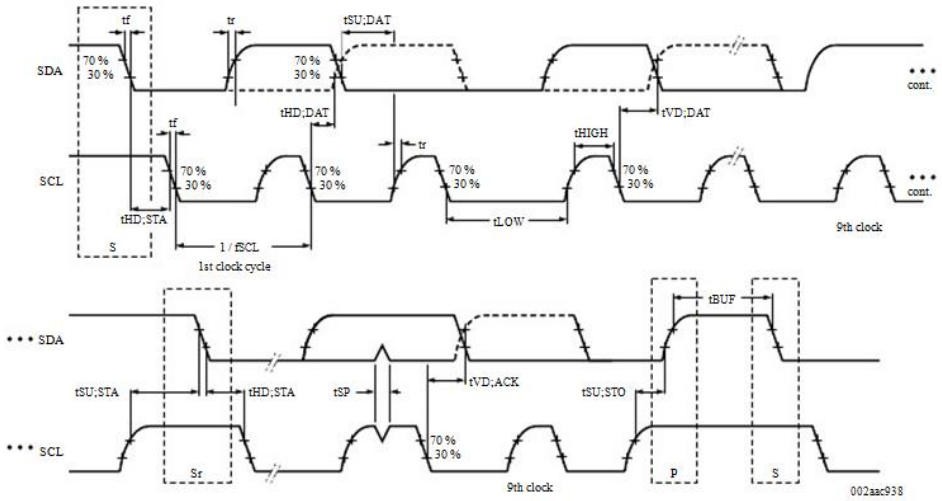


Figure 3-120 I2C AC Timing Diagram of F/S Mode

Table 3-107 I2C AC Timing Parameters for HS Mode

Symbol	Parameter	Cb = 100 pF (max)		Cb = 400 pF		Unit	Note
		Min	Max	Min	Max		
fSCL	SCL clock frequency	0	3.4	0	1.7	MHz	-
tSU;STA	Set-up time (repeated) START condition	160	-	160	-	ns	-
tHD;STA	Hold time (repeated) START condition	160	-	160	-	ns	-
tLOW	LOW period of the SCL clock	160	-	320	-	ns	-
tHIGH	HIGH period of the SCL clock	60	-	120	-	ns	-
tHD;DAT	Data hold time	0	70	0	150	ns	I2C-bus devices
tSU;DAT	Data set-up time	10	-	10	-	ns	-
tr	Rise time of SCLH signal	10	40	20	80	ns	-
tf	Fall time of SCLH signal	10	40	20	80	ns	
tSU;STO	Set-up time for STOP condition	160	-	160	-	ns	-

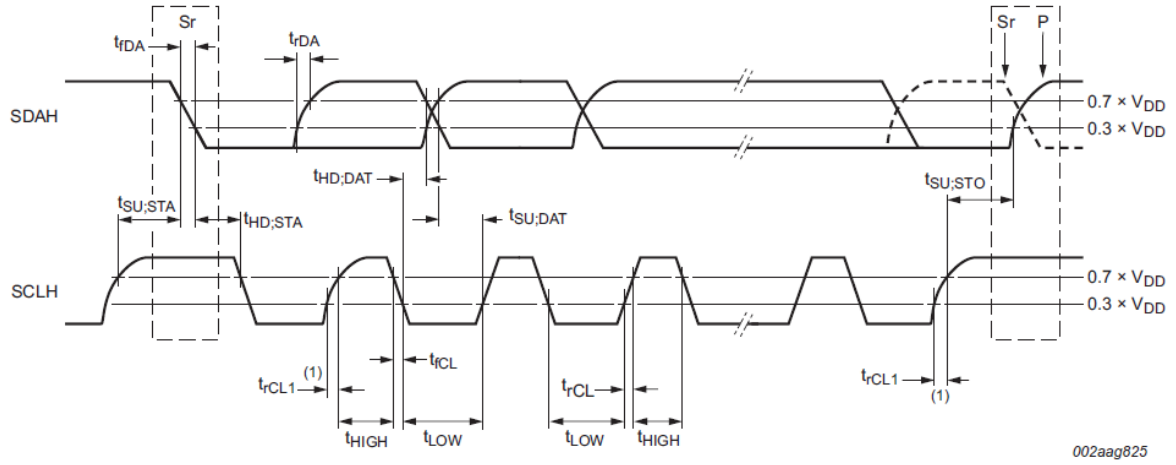


Figure 3-121 I2C AC Timing Diagram of HS Mode

For the I3C AC timing of the SDR mode, please refer to the MIPI I3C specification.

3.12.1.10 Programming Guide

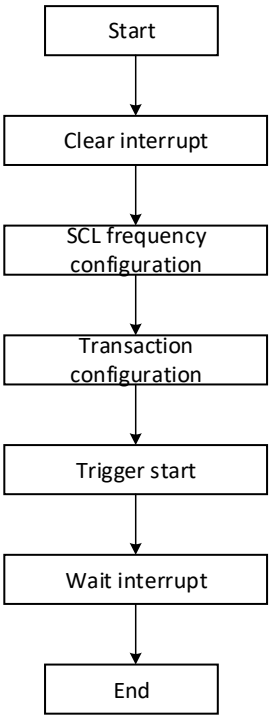


Figure 3-122 Programming Flow of I3C/I2C

Figure 3-122 shows the programming workflow of the I3C/I2C master. To initiate transmission of register data to the master, the AP must first clear the interrupt, and then configure the frequency division factor of the SCL to correspond to the low and high speeds (for the I2C HS mode or the I3C SDR mode). Then, the AP configures relevant registers for transmission such as the slave address, transfer length, transaction length. The transmission is then triggered by the Start register. Upon completion of the transfer or in the event of an error, an interrupt is generated, which should be cleared after the transmission is completed correctly or an error occurs.

3.12.2 Universal Asynchronous Receiver/Transmitter (UART)

3.12.2.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) is a full duplex serial communication channel between the chip and external devices. It is designed to be compatible with a range of standard software drivers and offer M16C450 and M16550A modes.

The UART supports word lengths from 5 to 8 bits, an optional parity bit, and one or two stop bits, all of which are fully programmable via the CPU interface. Additionally, the UART includes:

- A 16-bit programmable baud rate generator
- An 8-bit scratch register
- Modem CTS (Clear to Send) and RTS (Request to Send) control lines
- A diagnostic loopback mode

Furthermore, the UART is equipped with two Direct Memory Access (DMA) handshake lines for indicating when the FIFOs are ready to transfer data to the CPU, and can generate interrupts from any of these sources.

After the hardware reset, the UART is in the M16C450 mode; however, it can be enabled to enter the M16550A mode, which adds more advanced functions. These extensions are individually selectable via software control.

3.12.2.2 Features

- Provides 6 channels of UARTs
- UART1 and UART2 are 4-pin (TX, RX, CTS, RTS), while UART0, UART3, UART4, and UART5 are 2-pin channels (TX, RX) UART channels.
- Supports both M16C450 and M16550A operation modes
- Compatible with standard software drivers
- Transfer system: Asynchronous
- Data length: 5 to 8 bits
- Stop bits:1 or 2
- Programmable parity (even, odd, and no parity)
- Hardware flow control: CTS/RTS-based automatic transmission and reception control
- Software flow control: Uses special characters, XON and XOFF for software flow control
- Baud rate programmable up to 3 Mbps
- Baud rate error: Less than 0.25 %
- Baud rate accuracy compensation
- Interrupt request: Receive interrupts/transmit interrupts
- Maskable interrupts
- Two independent 32-depth FIFOs for transmit and receive
- Data transfer: Supports DMA (Transmit/Receive) transfer
- Escape character sequence detection

3.12.2.3 Block Diagram

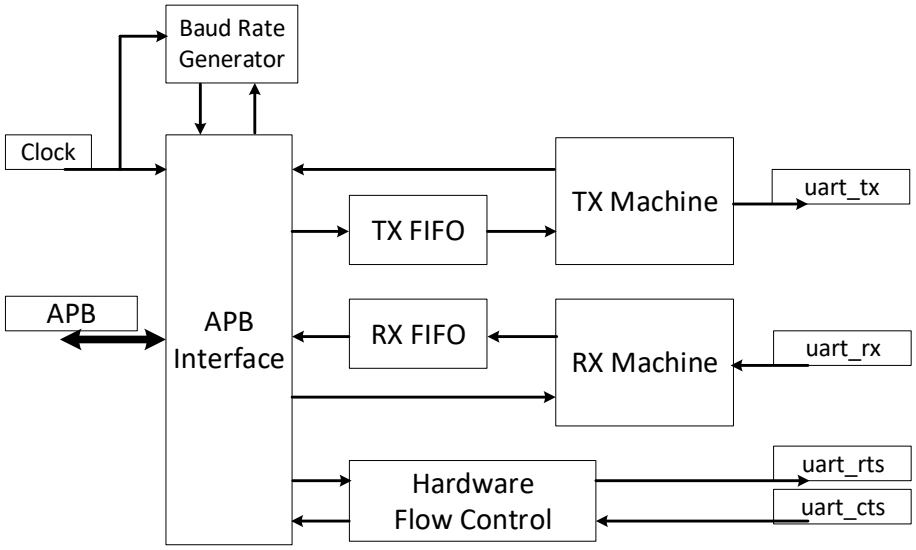


Figure 3-123 Block Diagram of UART

The figure above depicts the block diagram of UART, which consists of

- First In First Out (FIFO)
- Finite State Machine (FSM)
- Advanced Peripheral Bus (APB) interface
- Modem control

The UART supports full-duplex serial communication through its TX and RX channels, each of which contains an FSM (TX FSM or RX FSM) and a 32-byte FIFO (TX FIFO or RX FIFO). The FSM indicates the current transfer stage of the TX or RX channel, while the FIFOs store the data to be sent or received. The APB interface allows the system to:

- Access the configuration and status registers of UART
- Read the received data or write the data to be sent

The modem control enables the UART to support the hardware flow control.

3.12.2.4 Function Description

3.12.2.4.1 Signals

3.12.2.4.1.1 External Signals

Table 3-108 External Signals Descriptions

Signal Name	Type	Description	Ball Location
UART0			
URXD0	DI	UART0 receive data	AP10
UTXD0	DO	UART0 transmit data	AR10
UART1			
URXD1	DI	UART1 receive data	AP8
UTXD1	DO	UART1 transmit data	AR8
UCTS1	DI	UART1 clear to send (active low)	AT10
URTS1	DO	UART1 request to send (active low)	AM8
UART2			
URXD2	DI	UART2 receive data	AP24, AM16, AT14
UTXD2	DO	UART2 transmit data	AM25, AN16, AT12
UCTS2	DI	UART2 clear to send (active low)	AN25, AR16, AL9
URTS2	DO	UART2 request to send (active low)	AN24, AP16, AM10
UART3			
URXD3	DI	UART3 receive data	H33
UTXD3	DO	UART3 transmit data	G35
UART4			
URXD4	DI	UART4 receive data	H34
UTXD4	DO	UART4 transmit data	H32
UART5			
URXD5	DI	UART5 receive data	AM20, AM10
UTXD5	DO	UART5 transmit data	AN20, AL9

3.12.2.4.1.2 Internal Signals

3.12.2.4.1.3 Clock

The UART has two clock input ports: Peripheral Clock (PCLK) and Baud Clock (BCLK).

- The **PCLK** is the APB clock with its frequency determined by the APB.
- The **BCLK** is the UART function clock with two levels of clock sources available for the UART to select the adapted clock frequency, allowing flexible implementation of multiple UARTs.

3.12.2.4.1.4 Reset

The UART features one asynchronous reset input port, which can be triggered by hardware reset, power reset or software reset bit, which can reset the initial values of all UART registers, FIFOs, and state machines.

3.12.2.4.1.5 Advanced Peripheral Bus (APB)

The UART is connected to the APB as an APB slave, allowing all UART registers to be programmed or read via the APB.

- Data transmitted to the outside via the TX FIFO can be written by the APB.
- Data received from the outside can be read from the APB via the RX FIFO.

3.12.2.4.1.6 Direct Memory Access (DMA)

Table 3-109 DMA Signal Descriptions

Signal Name	Signal Type	Description
tx_dmareq	Output	TX DMA request signal to DMA for transmission data.
tx_dmaack	Input	TX DMA acknowledge from DMA for transmission data
rx_dmareq	Output	RX DMA request signal to DMA for received data.
rx_dmaack	Input	RX DMA acknowledge from DMA for received data

3.12.2.4.1.7 Interrupt

When there is an output interrupt request, poll the UART interrupt identification register for more information about the interrupt, as descripted in Section 3.12.2.4.4.

3.12.2.4.2 Communication Protocol

- The 1-bit start bit must be low.
- The data bit length is 5 to 8 bits.
- The parity check can be odd or even.
- The stop bit is 1 to 2 bit(s) long and must be high.

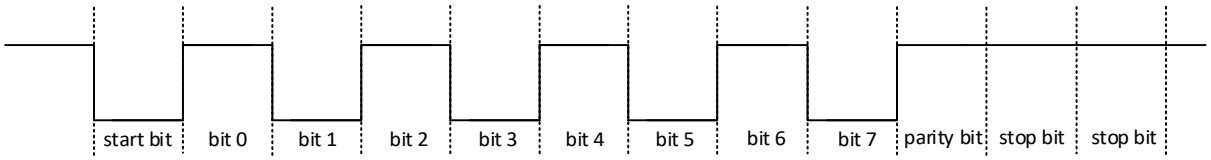


Figure 3-124 UART Communication Protocol

3.12.2.4.3 Clock Requirement

This section describes the clocks and special clocking requirements of the UART. For the definitions of PCLK and BCLK, please refer to Section 3.12.2.4.1.3.

- The **PCLK** is primarily used for programing registers, and also plays the roles below. The PCLK must be always running when the UART is in use.
  - The write clock of a TX FIFO
  - The read clock of an RX FIFO
  - Synchronization of FIFO.
- The **BCLK** is used for all state machines, writing RX FIFOs, and reading TX FIFOs. The BCLK must always be running when the UART is in use. To ensure a fixed baud rate when transmitting or receiving data, the BCLK must be kept in one clock source.

Note that for the 16x oversampling of input characters, the BCLK frequency is preferably greater than or equal to 16x the request baud rate. There is no special requirement for the PCLK.

3.12.2.4.4 Interrupt Definition

The UART generates several types of interrupts, as detailed in Table 3-110 and Table 3-111.

Table 3-110 UART Interrupt Control Bits and Interrupt Factors

UARTn+0004h Interrupt Enable Register UARTn_IER

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									CTSI	RTSI	XOFFI		EDSSI	ELSI	ETBEI	ERBFI
Type									R/W							
Reset									0							

IER[3:0] are modified when LCR[7] = 0.

IER[7:4] are modified when LCR[7] = 0&EFR[4] = 1.

UARTn+0008h Interrupt Identification Register UARTn_IIR

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									FIFOE		ID4	ID3	ID2	ID1	ID0	NINT
Type									RO							
Reset									0	0	0	0	0	0	0	1

ID4 and ID3 are presented only when EFR[4] = 1.

Table 3-111 UART Interrupt Types

Interrupt Type	Interrupt Request Bit (UARTn_IER)	Interrupt Identification (UARTn_IIR)	Interrupt Factor	Notes
Received	ERBFI	IIR[5:0] = 000100b	The RX buffer contains data.	-
	-	IIR[5:0] = 001100b	Timeout on the character in the RX FIFO	-
Transmit	ETBEI	IIR[5:0] = 000010b	The TX holding register is empty, or the contents of the TX FIFO have been reduced to the trigger level.	-
Communication Error	ELSI	IIR[5:0] = 000110b	If a frame error, parity error, break interrupt or FIFO overrun happens, the interrupt occurs.	For the detailed interrupt status, refer to LSR[4:1].
Modem	EDSSI	IIR[5:0] = 000000b	Modem status change	For the detailed interrupt status, refer to MSR[4:1].
Enhancement Feature	CTSI	IIR[5:0] = 100000b	If a rising edge is detected on the CTS, the interrupt occurs.	Available when the enhanced feature is enabled. (EFR[4] = 1)
	RTSI	IIR[5:0] = 100000b	If a rising edge is detected on the RTS, the interrupt occurs.	
	XOFF1	IIR[5:0] = 010000b	When an XOFF character is received, the interrupt occurs.	

3.12.2.4.5 Enhancement Features

The UART provides more enhanced features than the industry-standard 16550. This includes **hardware flow control**, **software flow control**, and the **escape** function.

3.12.2.4.5.1 Hardware Flow Control

The **hardware flow control** uses two dedicated signals, Clear to Send (CTS) and Request to Send (RTS) signals, to indicate whether the UART is ready to get data or send data.

This feature is highly advantageous in embedded application where the Interrupt Service Routine (ISR) latency is difficult to predict and control. The MCU is relieved from the requirement of fetching the received data within a fixed amount of time.

Table 3-112 CTS and RTS Signals Behaviors

Signal	Status	Description
CTS	Low	The UART can start to transmit data.
CTS	Active	The UART is not allowed to transmit data.
RTS	Low	The UART FIFO in the received circuit is sufficient to receive data.
RTS	High	The UART is not allowed to receive data.

3.12.2.4.5.2 Software Flow Control

The **software flow control** uses special programmable characters, XON and XOFF, to handle the flow of data. When XOFF is received, the UART transmission is halted and will not resume until XON is received.

Note:

- To enable any of the enhancement features, the enhanced mode bit EFR[4] must be set. If it is not set, IER[7:5], FCR[5:4], and MCR[7:6] cannot be written, ensuring the UART is backward compatible with software written for 16C450 and 16550A devices.
- When the oversampling ratio between the UART clock and baud rate is less than 8, it is necessary to enable the guard time function of the UART TX device to ensure the MediaTek UART RX works properly. Otherwise, frame errors could occur and the received data could get corrupted.

3.12.2.4.5.3 Escape Function

The **Escape** function can be enabled by configuring ESCAPE_EN and ESCAPE_DAT for the software flow control. The normal characters sent in the TX FIFO will be escaped to its inverse code if it matches XON/XOFF/ESCAPE_DAT and the transmit state machine will send ESCAPE_DAT before the inverse code. When the receive state machine receives the ESCAPE_DAT character, it will abandon it and escape the next character to its inverse code before saving it to the RX FIFO.

3.12.2.5 Theory of Operations

3.12.2.5.1 Register Remap

Table 3-113 UART Register Map

Address	Name	Description
UART_BASE+0x0C	LCR	Line Control Register (LCR)
UART_BASE+0x24	HIGHSPEED	HIGH SPEED UART
UART_BASE+0x28	SAMPLE_COUNT	SAMPLE_COUNT
UART_BASE+0x2C	SAMPLE_POINT	SAMPLE_POINT
UART_BASE+0x34	RATEFIX_AD	Rate Fix Address
UART_BASE+0x3C	GUARD	Guard Time Added Register
UART_BASE+0x40	ESCAPE_DAT	Escape Character register
UART_BASE+0x44	ESCAPE_EN	Escape Enable Register
UART_BASE+0x48	SLEEP_EN	Sleep Enable Register
UART_BASE+0x4C	DMA_EN	DMA Enable Register
UART_BASE+0x50	RXTRI_AD	RX Trigger Address
UART_BASE+0x54	FRACDIV_L	Fractional Divider LSB Address
UART_BASE+0x58	FRACDIV_M	Fractional Divider MSB Address

Address	Name	Description
UART_BASE+0x5C	FCR_RD	FIFO Control Register
UART_BASE+0x88	RTO_CFG	Rx Time Out Configuration Register
UART_BASE+0XB4	SLEEP_REQ	UART Sleep Request Register
UART_BASE+0xB8	SLEEP_ACK	UART Sleep Acknowledge Register

Table 3-114 UART Register Remap

Address	Name	Description	Name	Description
	Condition: LCR[7] == 0		**Condition: LCR[7] == 1**	
UART_BASE+0x00	THR RBR	TX Holding Register/ RX Buffer Register	DLL	Divisor Latch (LS)
UART_BASE+0x04	IER	Interrupt Enable Register	DLM	Divisor Latch (MS)
	Condition: LCR != 0xBF		**Condition: LCR == 0xBF**	
UART_BASE+0x08	FCR IIR	FIFO Control Register/ Interrupt Identification Register	EFR	Enhanced Feature Register
UART_BASE+0x10	MCR	Modem Control Register	XON1	XON1
UART_BASE+0x14	LSR	Line Status Register	XON2	XON2
UART_BASE+0x18	MSR	Modem Status Register	XOFF1	XOFF1
UART_BASE+0x1C	SCR	Scratch Register	XOFF2	XOFF2

3.12.2.5.2 Baud Rate Generation

The DLL (Divisor Latch LS (Least Significant Bit)) and DLM (Divisor Latch MS (Most Significant Bit)) act as clock dividers to obtain the required baud rate based on the BCLK frequency.

UARTn+0000h Divisor Latch (LS) UARTn_DLL

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									DLL[7:0]							
Type									R/W							
Reset									1							

UARTn+0004h Divisor Latch (MS) UARTn_DLM

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									DLM[7:0]							
Type									R/W							
Reset									0							

Note:

- The DLL and DLM can only be updated if LCR[7] is set (“1”). Note that the division by 1 generates a BAUD signal that is constantly high. The example below shows the divider that needs to generate a given baud rate from the clock inputs of 26 MHz.

When clock source = 26 MHz and baud rate = 4800 bps

If HIGHSPEED (0x24) = 0: 26 MHz/4800/16 ~= 339 = 0x153 → DLM: 0x01, DLL: 0x53

If HIGHSPEED (0x24) = 1: 26 MHz/4800/8 ~= 677 = 0x2A3 → DLM: 0x02, DLL: 0xA3

If HIGHSPEED (0x24) = 2: 26 MHz/4800/4 ~= 1354 = 0x54A → DLM: 0x05, DLL: 0x4A

If HIGHSPEED (0x24) = 3: 26 MHz/4800/256 + 1 ~= 22 = 0x16 → DLM: 0x00, DLL: 0x16

UARTn+0024h HIGH SPEED UART UARTn_HIGHSPEED

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name															SPEED [1:0]	
Type															R/W	
Reset															0	

Note:

SPEED UART sample counter base

0: Based on 16*baud_pulse, baud_rate = system clock frequency/16/{DLM, DLL}

1: Based on 8*baud_pulse, baud_rate = system clock frequency/8/{DLM, DLL}

2: Based on 4*baud_pulse, baud_rate = system clock frequency/4/{DLM, DLL}

3: Based on sample_count * baud_pulse, baud_rate = system clock frequency/(sample_count+1)/{DLM, DLL}

UARTn+0028h SAMPLE_COUNT UARTn_SAMPLE_COUNT

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									SAMPLECOUNT [7:0]							
Type									R/W							
Reset									0							

Note:

When HIGHSPEED = 3, the sample_count is the threshold value for the UART sample counter (sample_num).

Sample Count = clock source/baud rate/{DLM. DLL} - 1

For example, clock source: 26 MHz, baud rate: 4800 bps; DLM: 0x00, DLL: 0x16

If High Speed (0x24) = 0, 1 or 2: Unnecessary to set SAMPLE_COUNT

If High Speed (0x24) = 3: 26 MHz/4800/0x16 - 1 ~= 245 → SAMPLE_COUNT = 245

UARTn+002Ch SAMPLE_POINT UARTn_SAMPLE_POINT

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									SAMPLEPOINT [7:0]							
Type									R/W							
Reset									8'hff							

Note:

If HIGHSPEED = 3, UART gets the input data when sample_count = sample_num.

The SAMPLE_POINT is usually ROUNDDOWN ((SAMPLE_COUNT+1)/2 - 1).

For example, if the clock source = 26 MHz, the baud rate = 4800 bps,

DLL: 0x00, and DLL: 0x16&SAMPLE_COUNT = 245

SAMPLE_POINT = ROUNDDOWN ((245+1)/2 – 1) = 122 (sample the central point to decrease inaccuracy)

3.12.2.5.3 Accuracy Compensation

FRACDIV_L and FRACDIV_M act as baud rate accuracy compensation factors to reduce baud rate errors.

UARTn+0054h Fractional Divider LSB Address UARTn_FRACDIV_L

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									FRACDIV_L							
Type									R/W							
Reset									0	0	0	0	0	0	0	0

Note:

- FRACDIV_L: Add sampling count (+1) from state data7 to state data0 in order to improve fractional divisor.

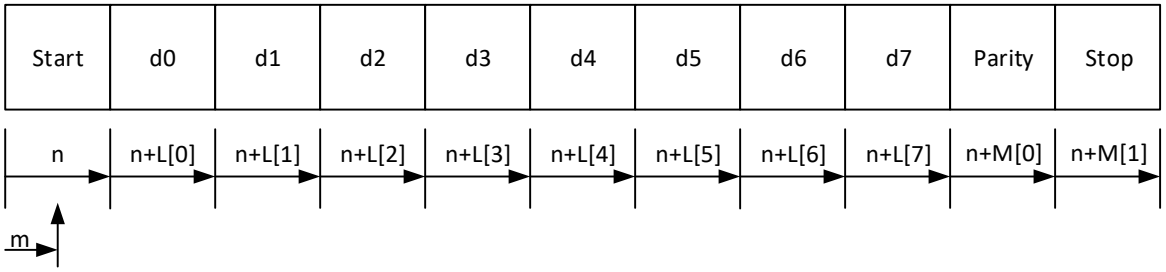
UARTn+0058h Fractional Divider MSB Address UARTn_FRACDIV_M

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name															FRACDIV_M	
Type															R/W	
Reset															0	0

Note:

- FRACDIV_M: Add sampling count in the stop state and the parity state to improve the fractional dividers.
- FRACDIV_L/FRACDIV_M: Add one sampling period to each symbol to increase baud rate accuracy.

bit_extend register = FRACDIV_L[7:0]
FRACDIV_M[1:0]



Bit extend number = ROUND ((clock source/baud rate/{DLM. DLL} – (SAMPLE_COUNT + 1))*10)
For example, if the clock source = 26 MHz, the baud rate = 4800 bps;
DLM: 0x00, DLL: 0x16, and SAMPLE_COUNT = 245.
Bit extend number = ROUND ((26MHz/4800/0x16 – (245+1))*10) = 2
Therefore, it should compensate for 2 bits of one frame (e.g. FRACDIV_L = 0x44, FRACDIV_M = 0x00). Refer to Table 3-115 for more details.

Table 3-115 Bit Extension Number Reference

Bit Extension Number	FRACDIV_M	FRACDIV_L
0	0x00	0x00
1	0x00	0x10
2	0x00	0x44
3	0x00	0x92

Bit Extension Number	FRACDIV_M	FRACDIV_L
4	0x01	0x29
5	0x01	0xaa
6	0x01	0xb6
7	0x01	0xdb
8	0x01	0xef
9	0x01	0xff
10	0x03	0xff

3.12.2.5.4 Data Transmission

See Table 3-116 for details of the data transmission mechanism.
The software can write transmitted data into the THR by asserting transmit interrupt (IIR[5:0] = 000010b) or polling the THRE bit status as “1” directly. When FIFOs are enabled, the transmitted data can be written into the THR. The data is transferred to the TX FIFO directly.

Table 3-116 Data Transmission Mechanism

FIFIO Status	Condition	Result
FIFO disabled	THR empty	THRE bit of LSR is 1. Transmitted data can be written in THR.
	THR not empty	TX starts to transmit automatically.
FIFO enabled	TX FIFO reduced to trigger level	THRE bit of LSR is 1. Transmitted data can be written in THR.
	TX FIFO not empty	TX starts to transmit automatically.

3.12.2.5.5 Data Reception

- When the RX Buffer is almost full or a byte is being transferred into the RX FIFO, the DR bit of the LSR is “1” and the received data can be read by the RX Buffer Register (RBR).
- The software can read the received data either by responding to a received interrupt (IIR[5:0] = 000100b), or by polling the DR bit status directly. If FIFOs are enabled, the received data in the RX FIFO can be read by reading RBR.

3.12.2.6 Power Management

Every UART has its own clock gating to save power. Enabling clock gating can turn off the clock once the transfer is complete with the UART. Furthermore, the UART BCLK and PCLK share the same clock gating bit.

In special scenarios where the system needs to enter the sleep mode while the UART is either transmitting or receiving character, the UART hardware provides *sleep_req* and *sleep_ack* mechanisms to avoid conflicts. Set *sleep_req* and then poll *sleep_ack* until its status becomes “1”, which indicates that the UART has entered an idle status (TX idle and RX idle).

- TX idle: The TX state machine enters an idle status.

- RX idle: The RX state machine enters an idle status and the RX data line remains idle (high) for the duration of five-bit baud.

3.12.2.7 Electrical Specifications

Table 3-117 UART Electrical Specifications

Symbol	Description	Min	Max	Unit
TX_ERR (1)	TX baud rate error rate (3000000, 2000000, 1000000, 921600, 460800, 230400, 115200, 76800, 57600, 38400, 28800, 19200, 9600)	-0.25	0.25	%
RX_ERR (2)	Acceptable error rate on RX baud rate (3000000, 2000000, 1000000, 921600, 460800, 230400, 115200, 76800, 57600, 38400, 28800, 19200, 9600)	-1.5	1.5	%

(1) TX_ERR (%): The percentage of ((real TX output baud rate – target baud rate)/target baud rate).

(2) RX_ERR (%): The acceptable error rate on the RX, i.e. the percentage of ((real RX acceptable baud rate – target baud rate)/target baud rate).

The figure below depicts the specific UART baud rate waveform.

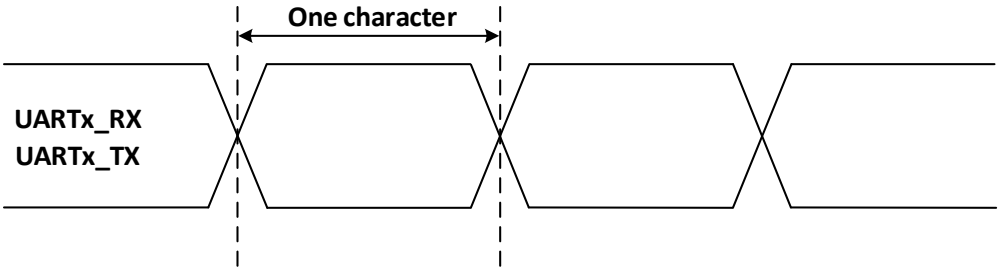


Figure 3-125 UART Timing Diagram

3.12.2.8 Programming Guide

3.12.2.9 Baud Rate Setting and UART Initialization

The table below suggests the UART baud rate setting from a clock input of 30 MHz.

Table 3-118 Suggested UART Baud Rate Setting

Baud Rate	HIGHSPEED	{DLM, DLL}	SAMPLE_COUNT	SAMPLE_POINT	FRACDIV_M	FRACDIV_L
3,000,000	3	0x00, 0x01	0x09	0x04	0x0	0x0
2,000,000	3	0x00, 0x01	0x0E	0x06	0x0	0x0
1,000,000	3	0x00, 0x01	0x1D	0x0E	0x0	0x0
500,000	3	0x00, 0x01	0x3B	0x1D	0x0	0x0
250,000	3	0x00, 0x01	0x77	0x3B	0x0	0x0
153,600	3	0x00, 0x01	0xC2	0x60	0x0	0x92
115,200	3	0x00, 0x02	0x81	0x40	0x0	0x44
76,800	3	0x00, 0x02	0xC2	0x60	0x0	0x92
57,600	3	0x00, 0x03	0xAC	0x55	0x1	0xB6

Baud Rate	HIGHSPEED	{DLM, DLL}	SAMPLE_COUNT	SAMPLE_POINT	FRACDIV_M	FRACDIV_L
38,400	3	0x00, 0x04	0xC2	0x60	0x0	0x92
28,800	3	0x00, 0x05	0xCF	0x67	0x0	0x92
19,200	3	0x00, 0x07	0xDE	0x6E	0x0	0x44
9,600	3	0x00, 0x0D	0xEF	0x77	0x1	0x92
7,200	3	0x00, 0x11	0xF4	0x79	0x1	0x01
4,800	3	0x00, 0x19	0xF9	0x7C	0x0	0x0

Use the registers below to set the baud rate.

- DLL
- DLM
- HIGHSPEED
- SAMPLE_COUNT
- SAMPLE_POINT
- FRACDIV_M
- FRACDIV_L

Once the baud rate is set, the UART is able to start transmitting data by filling the TX FIFO and receiving data from the RX FIFO. See Table 3-119 for an example of how to set the baud rate to 115,200 bps using a clock input of 30 MHz.

Table 3-119 UART Baud Rate Setting Example

Step	Description	Related Register Setting
1	Select UART sample counter base to SPEED 3	HIGHSPEED = 0x3
2	Set sample counter.	SAMPLE_COUNT = 0x81 SAMPLE_POINT = 0x40 FRACDIV_L = 0x44 FRACDIV_M = 0x0
3	Switch register to the divisor mode (Register MAP condition 2) to execute the divisor latch setting. UARTn_LCR[7] = 1	LCR = 0x80
4	Set the divisor latch.	DLL = 0x2 DLM = 0x0
5	Set the guard time.	GUARD
6	Switch the register to the normal mode (Register MAP condition 1). UARTn_LCR[7] = 0	LCR = 0x00

Table 3-120 UART Hardware Initialization

Step	Description	Related Register Setting
1	Set the baud rate (refer to Table 3-119).	-
2	Enable the enhanced features. (The register is accessible only when LCR = 0xBF.)	LCR = 0xBF EFR = 0x10 LCR = 0x00

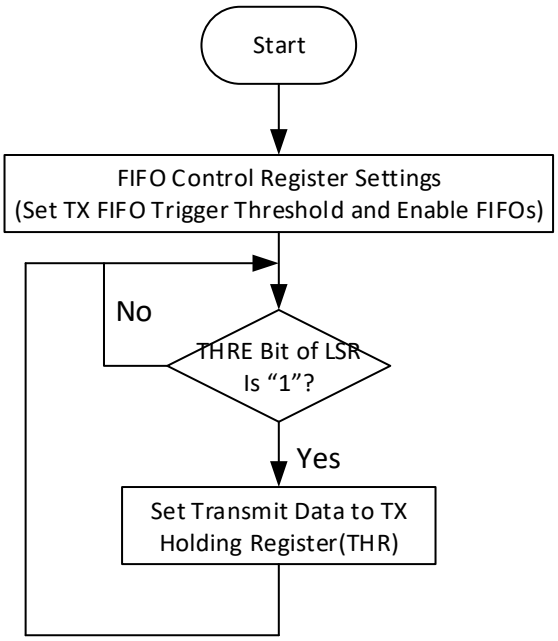
Step	Description	Related Register Setting
3	Enable the FIFO control.	FCR
4	Set word length (LCR[1:0]), parity (LCR[5:4]) and STOP (LCR[2]) bits.	LCR
5	Enable interrupts.	IER

The suggested software programming sequence is as follows:

```
1    DRV_WriteReg32(UART_BASE+0x24, 0x00000003); //High-speed UART
2    DRV_WriteReg32(UART_BASE+0x28, 0x00000081); //sample_count
3    DRV_WriteReg32(UART_BASE+0x2C, 0x00000040); //sample_point
4    DRV_WriteReg32(UART_BASE+0x4C, 0x00000001); //Enable RX DMA
5    DRV_WriteReg32(UART_BASE+0x54, 0x00000044); //FRACDIV_L
6    DRV_WriteReg32(UART_BASE+0x58, 0x00000000); //FRACDIV_M
7    DRV_WriteReg32(UART_BASE+0x0C, 0x000000BF); //LCR==0xBF, change to Condition 2
8    DRV_WriteReg32(UART_BASE+0x00, 0x00000002); //DLL, LS
9    DRV_WriteReg32(UART_BASE+0x04, 0x00000000); //DLM, MS
10   DRV_WriteReg32(UART_BASE+0x08, 0x00000010); //Enable enhancement features
11   DRV_WriteReg32(UART_BASE+0x0C, 0x00000000); //LCR !=0xBF, change to Condition 1
12   DRV_WriteReg32(UART_BASE+0x08, 0x00000031); //FIFO trigger threshold and enable FIFO
13   DRV_WriteReg32(UART_BASE+0x0C, 0x00000003); //8-bit word length
14   DRV_WriteReg32(UART_BASE+0x04, 0x00000001); //Enable RX interrupt
```

3.12.2.10 Data Transmission and Reception

3.12.2.10.1 Data Transmission

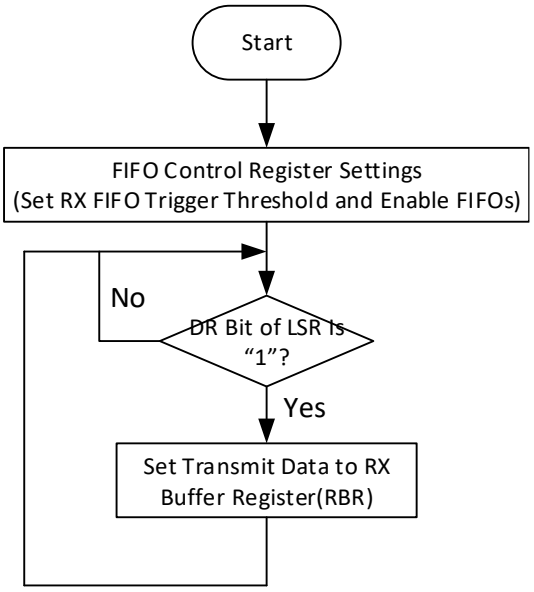


FIFO register settings:

- Enable FIFO: Set 0x08[0] to 1’b1
- TX FIFO Trigger Threshold 0x08[5:4]
- 2’b00: 1 byte (down to trigger)
- 2’b01: 4 bytes (down to trigger)
- 2’b10: 8 bytes (down to trigger)
- 2’b11: 14 bytes (down to trigger)

Figure 3-126 UART Data Transmission with THRE Bit Status Polling

3.12.2.10.2 Data Reception



FIFO register settings:

- Enable FIFO: Set 0x08[0] to 1'b1
- RX FIFO Trigger Threshold 0x08[7:6]
- 2'b00: 1 byte (up to trigger)
- 2'b01: 6 bytes (up to trigger)
- 2'b10: 12 bytes (up to trigger)
- 2'b11: register 0x50[4:0] (up to trigger)

Figure 3-127 UART Data Reception with DR Bit Status Polling

3.12.2.10.3 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.12.2.10.4 Reference

16550 UART

3.12.3 Infrared Receiver (IRRX)

The device is equipped with one IRRX module, which serves as a receiver for infrared (IR) assistive applications.

3.12.3.1 IRRX Overview

The IRRX module can receive IR signals and support IR protocols including NEC, RC5, RC6 and RCMM (Remote Control Multimedia) protocols. The IR transmission protocols are categorized into two types:

- Pulse-width coding transmission protocols, such as NEC and RCMM protocols. See Figure 3-128.
- Bi-phase coding transmission protocols, such as RC5 and RC6 protocols. See Figure 3-129. The IRRX module can decode the signal by a constant period sample pulse.

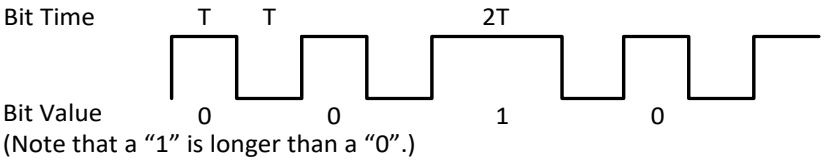


Figure 3-128 Pulse-width Coding

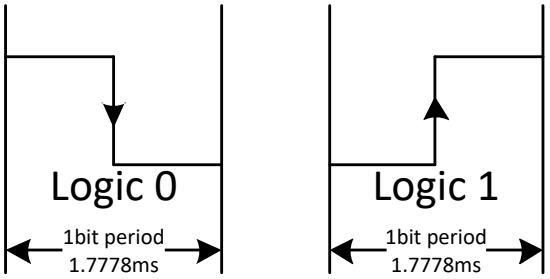


Figure 3-129 Bi-phase Coding

3.12.3.2 IRRX Features

The IRRX module supports the following key features:

- Hardware decoding support for certain IR transmission protocols:
  - Pulse width coding (NEC, RCMM)—decoding the IR signal by a variable-period sampling pulse
  - Bi-phase coding (RC5, RC6)—decoding the IR signal by a constant-period sampling pulse with inversion in phase
- Software decoding support for other protocols
- Wake-up interrupt generation in hardware decoding mode

3.12.3.3 IRRX Block Diagram

Figure 3-130 shows the block diagram of IRRX, which includes both software and hardware decoders.

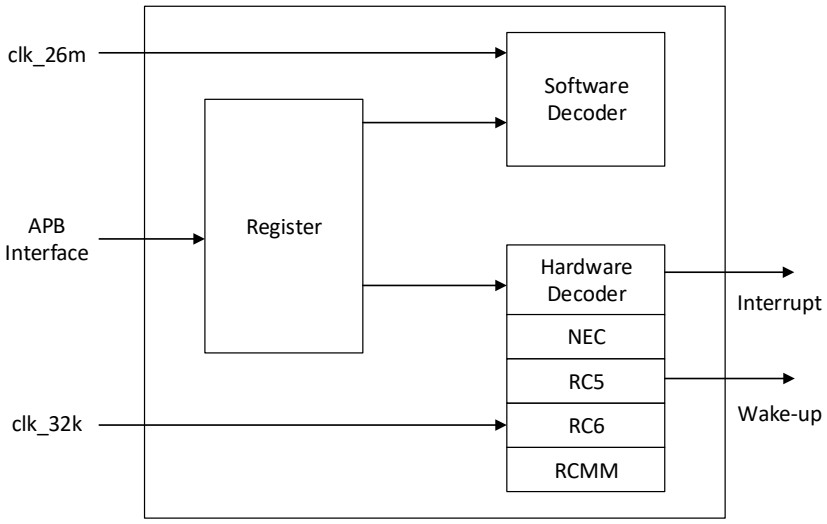


Figure 3-130 Block Diagram of IRRX

3.12.3.4 IRRX Function Description

IRRX has two decoding modes: hardware decoding and software decoding. IRRX hardware decoding can decode the input waveform of NEC, RCMM, RC5 and RC6 protocols. IRRX software decoding can decode the input waveform of NEC, RCMM, RC5 and RC6 protocols and any other protocols through software.

IRRX hardware decoding also supports the system wake-up function. There are two ways to trigger a wake-up: by a specific key or by any key.

3.12.3.4.1 IRRX Hardware Decoding

IRRX hardware decoding can decode the input waveform of NEC, RCMM, RC5 and RC6 protocols through hardware circuits. IRRX module records the decoded value in register PDREG_IRM(0x004) and register PDREG_IRL(0x008), with the decoded value bit being up to 56. The decoded value can be read through APB when a decoding end interrupt asserts. The working frequency of hardware decoding is 32 kHz.

3.12.3.4.2 IRRX Software Decoding

IRRX software decoding can decode the input waveform of NEC, RCMM, RC5 and RC6 protocols and any other protocols through software. In IRRX software decoding mode, hardware circuits record the input pulse width in register CHK_DATA0(0x0A0) to register CHK_DATA31(0x11C). The recorded pulse width can be up to 124 and each pulse width length is 1 byte. The software can decode any input waveform by these recorded pulse width values. The working frequency of software decoding is 26 MHz.

3.12.3.4.3 IRRX Wake-up Function

In hardware decoding mode, IRRX module supports system wake-up function. There are two ways to trigger a wake-up:

- **Wake-up by a specific key:** If both the decode value and the number of bits match the value configured by the register, a wake-up flag signal is generated at the end of each transmission.
- **Wake-up by any key:** A wake-up flag is generated whenever a valid protocol transfer ends.

3.12.3.5 IRRX Theory of Operations

3.12.3.5.1 Hardware Decoding

In IRRX hardware decoding mode, hardware circuits decode input pulse values directly. The most important parameter of the hardware decoding is SA_PERIOD (hardware decoding sampling period, REG 0x10[7:0]). For NEC protocol, SA_PERIOD*(1/32 kHz) is typically configured to be equal to the low pulse width; then, the input pulse value can be decoded as 0 or 1 according to pulse widths.

3.12.3.5.2 Software Decoding

In IRRX software decoding mode, hardware circuits do not decode input pulse values directly. It records the input pulse width values, including high pulse and low pulse values, in register CHK_DATA0(0x0A0) to register CHK_DATA31(0x11C). The recorded pulse width can be up to 128, with each pulse width length being 1 byte. The working frequency of software decoding is 26 MHz.

The software decoding needs to set two important parameters, CHK_EN (software decoding enable bit, REG 0xc[13]) and CHK_PERIOD (software decoding sampling period, REG 0x10[20:8]). CHK_PERIOD = 13'h160 (d'352) means the sampling period is (CHK_PERIOD+1)*T (26 MHz) = 353*(1/26 MHz)=13.5769us. If the low pulse width is 1250us, then the software decoding value is 0x5c, 13.5769us* 0x5c = 13.5769us* 92 = 1249.07us.

3.12.3.6 IRRX Signal Descriptions

Table 3-121 presents IRRX signal descriptions.

Table 3-121 IRRX Signal Descriptions

Signal Name	Type	Description	Ball Location
IR_IN	DI	Infrared receiver module input	D37, AM26

3.12.3.7 IRRX Programming Guide

Table 3-45 IRRX Decode Programming Flow

Step	Address	Register name	Local address	R/W	Default value	Description
Set IRRX control registers						
1	IRRX base address + 0x0c	PDREG_IRCFGH		RW	20'hF0000	Configure IRRX
2	IRRX base address + 0x10	PDREG_IRCFGL		RW	8'hff	Configure IRRX
3	IRRX base address + 0x14	PDREG_IRTHD			12'h600	Configure IRRX
Enable IRRX and wait for an IRRX interrupt						
4	IRRX base address + 0x0c	PDREG_IRCFGH	IREN	RW	1'b0	Set IREN = “1” to enable IRRX
5	IRRX base address + 0x84	PDREG_INTEN		RW	1'b0	Set INT_EN = “1” to enable IRRX interrupts
6						Wait for IRRX IRQ
Read the IRRX decode value and clear the IRRX interrupt						
7	IRRX base address + 0x00	PDREG_IRH		RO	32'h0	Read the IRRX decode value
8	IRRX base address + 0x04	PDREG_IRM		RO	32'h0	Read the IRRX decode value
9	IRRX base address + 0x08	PDREG_IRL		RO	32'h0	Read the IRRX decode value
10	IRRX base address + 0x88	PDREG_IR_INTCLR		WO		Clear the IRRX interrupt

3.12.4 Serial Peripheral Interface (SPI)

3.12.4.1 SPI Master (SPIM)

3.12.4.1.1 SPI Master Overview

The Serial Peripheral Interface Master (SPIM) is a four-pin synchronous serial interface used for short-distance communication, primarily in embedded systems. The device features six SPIM controllers.

3.12.4.1.2 SPI Master Features

The SPIM supports the following key features:

- SPIM[0-3] support up to 52 MHz
- SPIM[4-5] support up to 27 MHz
- Two configurable transmit modes:
  - TX DMA mode—the SPI controller automatically fetches the transmitted data (to be put on the MOSI line) from memory
  - TX FIFO mode—the data to be transmitted on the MOSI line is written to FIFO before the start of the transaction.
- Two configurable receive modes:
  - RX DMA mode—the SPI controller automatically stores the received data (from MISO line) to memory.
  - RX FIFO mode—the received data is kept in RX FIFO of the SPI controller. The processor must read back the data by itself.
- Configurable chip-select setup, hold, and idle times
- Programmable serial clock high and low times
- Configurable transmit and receive bit order (MSB or LSB)
- Adjustable endian order from/to memory system
- Programmable byte length for transmission
- Unlimited length for transmission using dedicated pause mode
- Configurable option to control chip-select de-assertion between byte transfers
- Supports all clock polarity and phase modes
- SPIM signal descriptions

3.12.4.1.3 SPIM Block Diagram

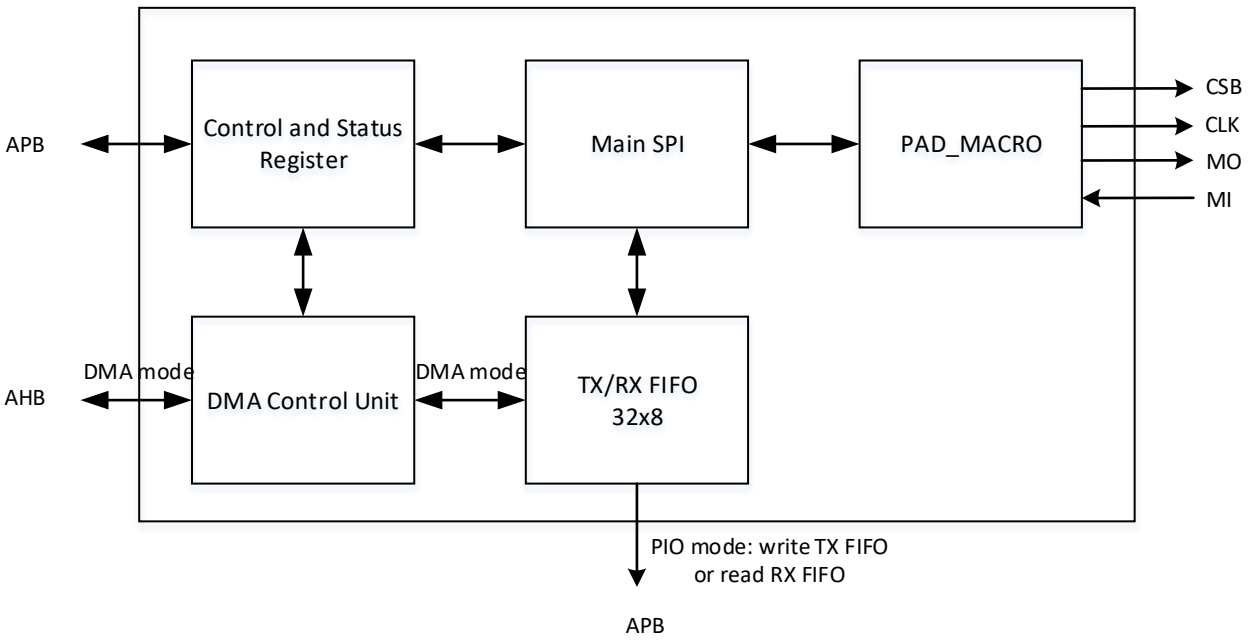


Figure 3-131 Block Diagram of SPIM

The SPIM consists of the following modules:

Table 3-122 SPIM Signal Descriptions

Module	Description
Control and status register	Receives commands from the system
DMA control unit	Communicate with SYSRAM when the SPI is set to the DMA mode
Main SPI	The functional unit
TX/RX FIFO	Both TX and RX have 32 x 8 bits FIFO for data storage
PAD_MACRO	Controls the SPI data capture and transmission to/from the SPI

- In the **PIO mode**, software can write data into the TX FIFO via *SPI_TX_DATA*, or read data from the RX FIFO via *SPI_RX_DATA.*
- In the **DMA mode**, the SPI is capable of automatically retrieving data from or sending data to SYSRAM via the AHB after the software configures the DMA parameters.

3.12.4.1.4 SPI Master Signal Descriptions

Table 3-123 presents SPIM signal descriptions.

Table 3-123 SPIM Signal Descriptions

Signal Name	Type	Description	Ball Location
SPI0 Master			
SPIM0_CLK	DO	SPIM0 clock	F32
SPIM0_CSB	DO	SPIM0 chip select	F31
SPIM0_MI	DI	SPIM0 data in	G31
SPIM0_MO	DO	SPIM0 data out	G32
SPI1 Master			
SPIM1_CLK	DO	SPIM1 clock	AU23
SPIM1_CSB	DO	SPIM1 chip select	AT22
SPIM1_MI	DI	SPIM1 data in	AR23
SPIM1_MO	DO	SPIM1 data out	AT23
SPI2 Master			
SPIM2_CLK	DO	SPIM2 clock	AP22
SPIM2_CSB	DO	SPIM2 chip select	AN23
SPIM2_MI	DI	SPIM2 data in	AP23
SPIM2_MO	DO	SPIM2 data out	AN22
SPI3 Master			
SPIM3_CLK	DO	SPIM3 clock	AT21
SPIM3_CSB	DO	SPIM3 chip select	AU21
SPIM3_MI	DI	SPIM3 data in	AN21
SPIM3_MO	DO	SPIM3 data out	AP21
SPI4 Master			
SPIM4_CLK	DO	SPIM4 clock	AN14
SPIM4_CSB	DO	SPIM4 chip select	AL14
SPIM4_MI	DI	SPIM4 data in	AK14
SPIM4_MO	DO	SPIM4 data out	AM14

Signal Name	Type	Description	Ball Location
SPI5 Master			
SPIM5_CLK	DO	SPIM5 clock	AU15
SPIM5_CSB	DO	SPIM5 chip select	AK11
SPIM5_MI	DI	SPIM5 data in	AK10
SPIM5_MO	DO	SPIM5 data out	AR14

3.12.4.1.5 SPI Master Function Description

As introduced in Section 3.12.4.1.2, the following table extends further regarding the SPI features.

Table 3-124 SPIM Function Descriptions

Features	Description
IO speed	SPIM[0-3] support up to 52 MHz. SPIM[4-5] support up to 27 MHz.
Two configurable TX/RX modes: TX/RX DMA mode and TX/RX FIFO mode	• TX modes(1) – TX DMA mode(2): The SPI controller automatically fetches the transmitted data (to be put on the MOSI line) from memory. – TX FIFO mode: Data transmitted on the MOSI line is written to FIFO before the transaction starts. • RX modes(3) – RX DMA mode: The SPI controller automatically stores the received data (from the MISO line) to memory. – RX FIFO mode: Data received is stored in the RX FIFO of the SPI controller. The processor itself must read back the data.
Configurable chip-select setup time, hold time and idle time	Refer to Figure 3-132.
Configurable option to control CS_N de-assert between byte transfers	Refer to Figure 3-134.
The depth of the TX and RX FIFO is 32 bytes.	• TX: – In the TX DMA mode, data on the MOSI line is prepared priorly in memory(4), and the SPI controller automatically reads the data. – In the TX PIO mode, writing the *SPI_TX_DATA* register means to write 4 bytes to TX FIFO, whose pointer automatically moves towards the next 4 bytes. • RX: – In the RX DMA mode, data on the MISO line is automatically moved by the SPI controller. – In the RX PIO mode, reading from the *SPI_RX_DATA* register means to read 4 bytes from the RX FIFO, whose pointer automatically moves towards the next 4 bytes.
Programmable serial clock high time and low time	The high and low time can be set separately. Thus, for a given baud rate, the serial clock with a wide range of a duty cycle is generated.

Features	Description
Programmable byte length for transmission	• PACKET_LENGTH defines the number of bytes in one packet. The number of bytes in one packet = PACKET_LENGTH+1. • PACKET_LOOP_CNT defines the number of packets within one transaction. The number of packets in one transaction = PACKET_LOOP_CNT+1. • Total bytes of one transaction = (PACKET_LENGTH+1) * (PACKET_LOOP_CNT+1)
Unlimited length for transmission using the dedicated pause mode	Achieved by the pause mode operation. In this mode, the CS_N signal is always active (in other modes, normally low) after the transmission. At this time, the SPI controller is in the PAUSE_IDLE state and ready to receive the resume command. Refer to Figure 3-133 for the state transition.
Supports all clock polarity and phase modes	• There are four communication modes: mode 0/1/2/3 (Figure 3-135). • The modes define the SCLK edge on which the MOSI line toggles and the master samples to the MISO line. The modes also define the SCLK steady level: clock/high/low, when the clock is inactive. • Each mode is formally defined with a pair of parameters namely “Clock Polarity (CPOL)” and “Clock Phase” (CPHA).

(1) The value of SPI_TX_SRC must be 4-byte aligned.

(2) TX data must always be prepared before a transaction. In the DMA mode, TX_DMA_EN must be set to 1'b1. In the PIO mode, software must put data into the TX FIFO through the SPI_TX_DATA register.

(3) The value of SPI_RX_DST must be 4-byte aligned.

(4) For theory of operations, please refer to Section 3.12.4.1.6.

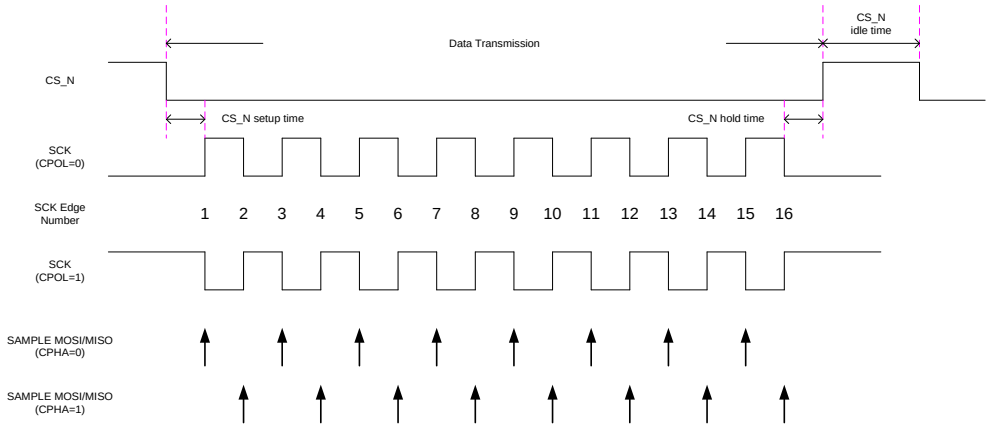


Figure 3-132 SPI Transmission Formats

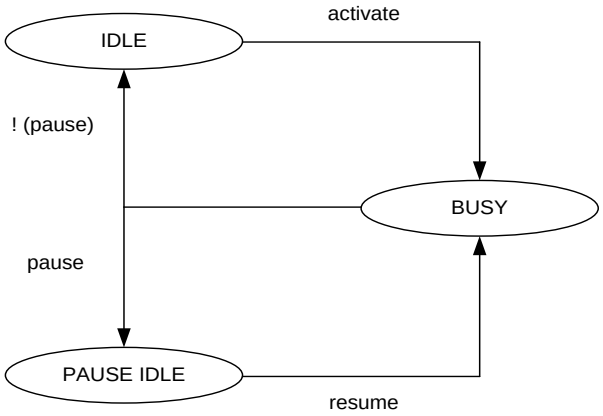


Figure 3-133 SPI Pause Mode State Transition

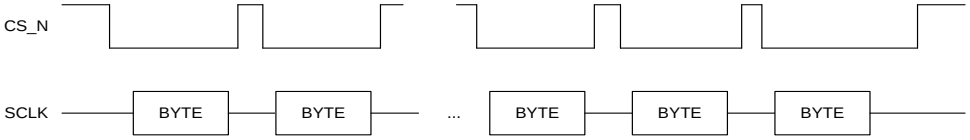


Figure 3-134 CS_N De-assert Mode

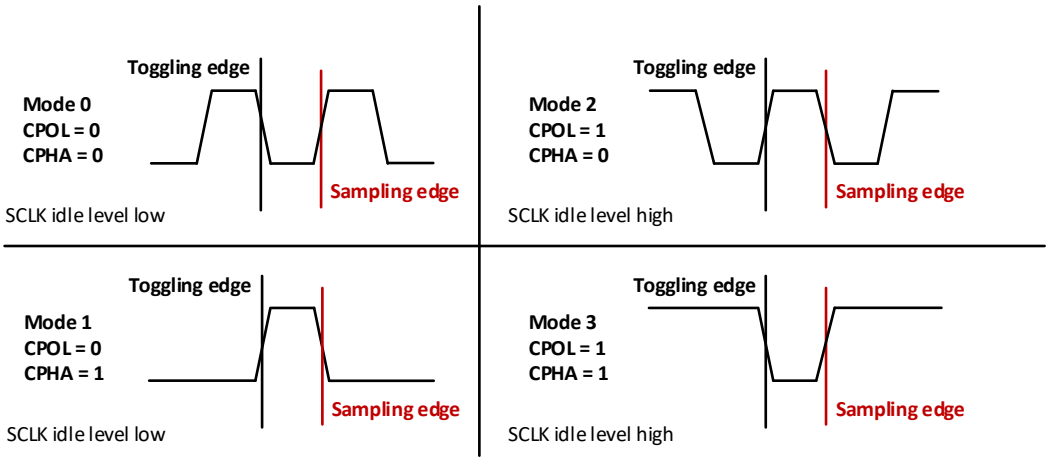


Figure 3-135 Four Communication Modes Waveform for SPI

3.12.4.1.6 SPI Master Theory of Operations

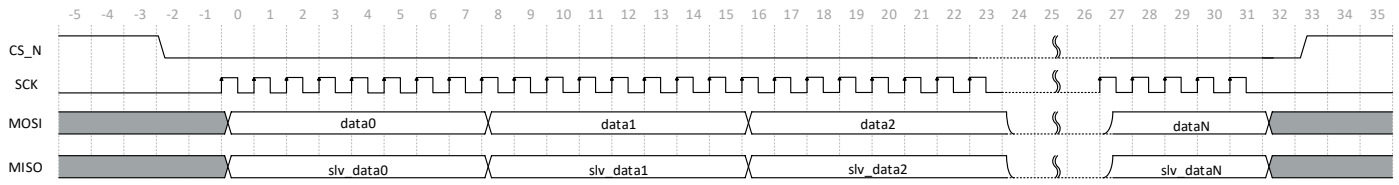


Figure 3-136 SPI Transaction Format

- In the DMA mode, data to be transferred should be prepared in SYSRAM in advance.
- In the PIO mode, the system must first push the data to be transferred into the SPI TX FIFO. Once the *START* command is received, the SPI sends the data to the slave continuously while simultaneously receiving data from the slave.

3.12.4.1.7 SPIM Timing Characteristics

Table 3-125 presents timing characteristics for the SPIM in the device.

Table 3-125 SPIM Timing Characteristics

No.	Parameter		Min	Typ	Max	Unit
	f_{OP_MCK}	SPI clock frequency			52	MHz
SPI02	t_c	Cycle time, SPI clock (SPI_CLK)	19.23(1)			ns
SPI05	$t_{w\_CLK\_L}$	Pulse duration, SPI_CLK low	7.2			ns
SPI06	$t_{w\_CLK\_H}$	Pulse duration, SPI_CLK high	7.2			ns
SPI07	t_{su_cs}(4)	SPI_CSB falling to SPI_SCK rising setup time	1.8			ns
SPI08	t_{h_cs}(4)	SPI_SCK falling to SPI_CSB rising hold time	1.8			ns
SPI09	t_{su_MOSI}	SPI_MO to SPI_CK rising setup time	6.6			ns
SPI10	t_{h_MOSI}	SPI_SCK rising to SPI_MO hold time	6.6			ns
SPI11	t_{su_MISO}(2)	SPI_MI to _SPI SCK rising setup time requirement	0			ns
SPI12	t_{h_MISO}(3)	SPI_SCK rising to SPI_MI hold time requirement	0			ns

(1) For maximum operating clock frequency, refer to Table 6-1.

(2) To achieve the minimum value of t_{su_MISO}, the internal sample clock delay of SPIM should be adjusted.

(3) t_{h_MISO} data valid time should be one cycle of f_{OP_MCK}.

(4) In CS GPIO mode, SPI_CSB is handled by the software. The software should pull down SPI_CSB pin before SPI starts transferring and pull up SPI_CSB pin when SPI completes the transaction.

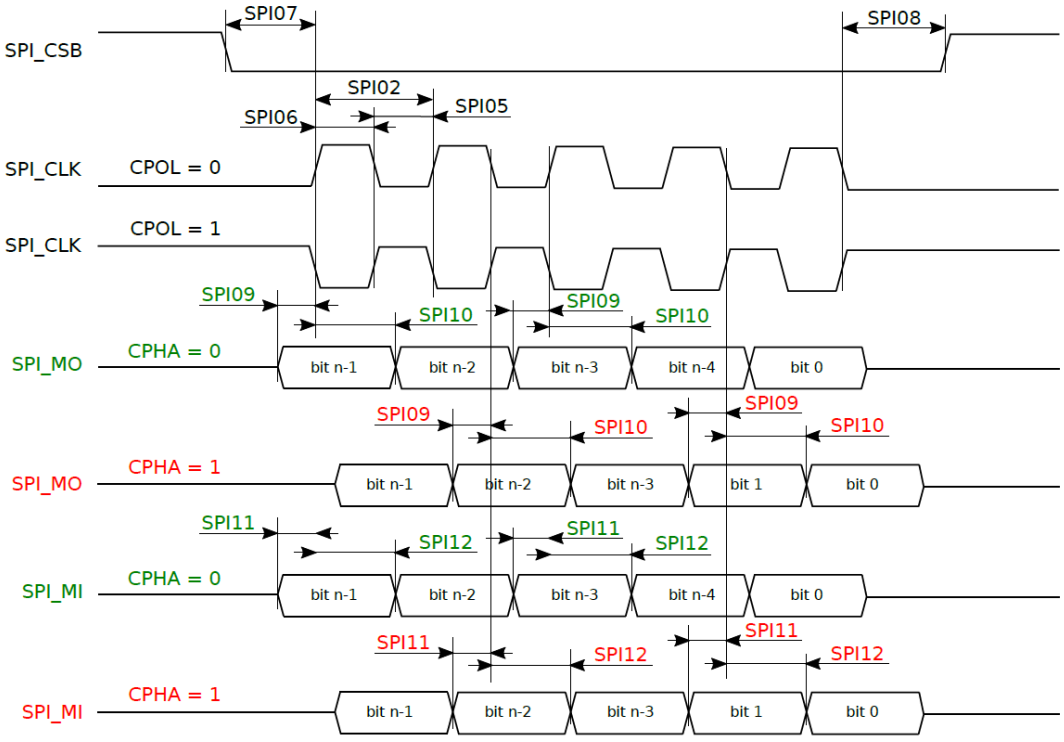


Figure 3-137 SPIM Timing Diagram

3.12.4.1.8 SPIM Programming Guide

Follow the steps below to perform SPI transmission:

1. Prepare the data in the memory with its start address to be the “source address”.
2. Set the timing and protocol for the SPI transmission (see Figure 3-133 for detailed setup parameters).
3. Fill in the “destination address”, which is the start address to place the received data, and “source address”, which is the start address to place the data to be transmitted, into the registers SPI_RX_DST and SPI_TX_SRC, respectively.
4. Write 1 to CMD_ACT to start the transfer.
5. Get the data received from the buffer prepared starting from “destination address”.

3.12.4.2 SPI Slave (SPIS)

3.12.4.2.1 SPI Slave Overview

The SPI Slave (SPIS) is a four-pin synchronous serial interface used for short-distance communication, primarily in embedded systems. The device features two SPIS controllers.

3.12.4.2.2 SPIS Features

The SPIS supports the following key features:

- SPIS[0-1] support up to 26 MHz
- Configurable transmit and receive bit order (MSB or LSB)
- Four SPI communication modes (M0, M1, M2, and M3)
- Enable or disable transmit and receive modes
- DMA mode programmable byte length:
  - TX DMA: from 1 byte to 1MB
  - RX DMA: from 1 byte to 1MB - 4 bytes
- Interrupt support
- TX FIFO and RX FIFO depths: 32 × 4 bytes

3.12.4.2.3 SPIS Block Diagram

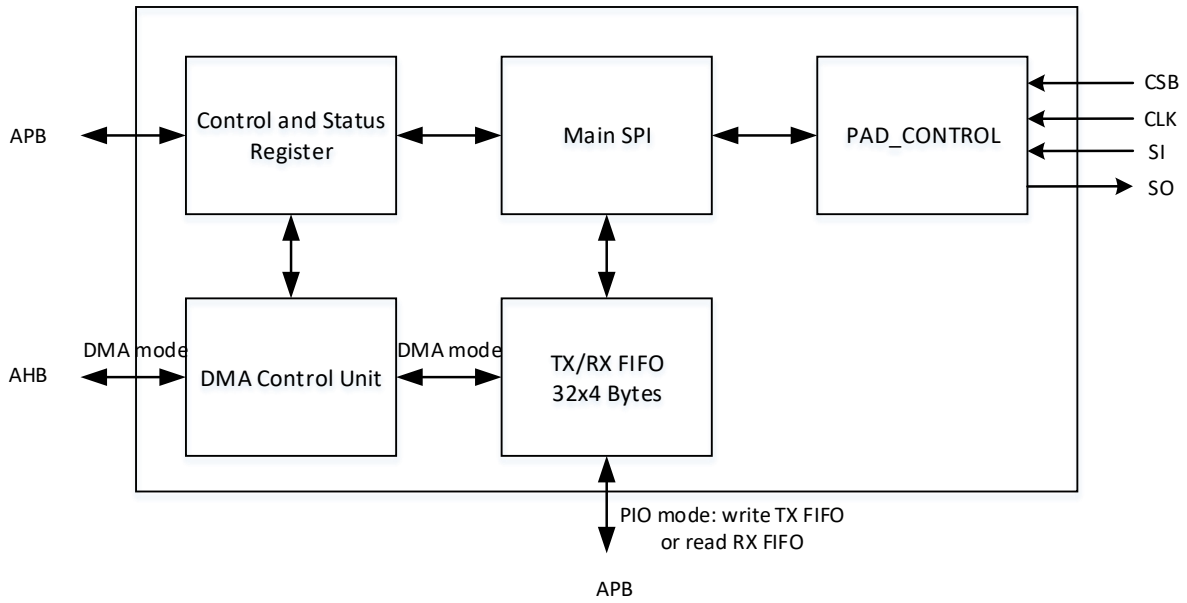


Figure 3-138 Block Diagram of SPIS

The SPI slave consists of the following modules:

Table 3-126 SPIM Signal Descriptions

Module	Description
Control and status register	Receives commands from the system
DMA control unit	Communicate with SYSRAM when the SPI is set to the DMA mode
Main SPI	The functional unit
TX/RX FIFO	Both TX and RX have 32 x 4 bytes FIFO for data storage
PAD_CONTROL	Controls the SPI data capture and data transmission to/from the SPI bus

- In the **PIO mode**, software writes data into TX FIFO via SPIS_TX_DATA or reads data from RX FIFO via SPIS_RX_DATA.
- In the **DMA mode**, the SPI is capable of automatically getting/sending data from/to SYSRAM via AHB after the software configures the DMA parameters.

3.12.4.2.4 SPIS Signal Descriptions

Table 3-127 presents SPIS signal descriptions.

Table 3-127 SPIS Signal Descriptions

Signal Name	Type	Description	Ball Location
SPI0 Slave			
SPIS0_CLK	DI	SPIS0 clock	F32
SPIS0_CSB	DI	SPIS0 chip select	F31
SPIS0_SI	DI	SPIS0 data in	G32
SPIS0_SO	DO	SPIS0 data out	G31
SPI1 Slave			
SPIS1_CLK	DI	SPIS1 clock	AU23
SPIS1_CSB	DI	SPIS1 chip select	AT22
SPIS1_SI	DI	SPIS1 data in	AT23
SPIS1_SO	DO	SPIS1 data out	AR23

3.12.4.2.5 SPIS Function Description

As introduced in Section 3.12.4.2.2, the following table extends further regarding the SPI features.

Table 3-128 SPIS Function Descriptions

Features	Description
IO speed	SPIM[0-1] support up to 26 MHz.
Configurable bit transmitting and receiving order	Two options of bit order – MSB or LSB first
Four SPI communication modes (M0, M1, M2, and M3)	• The modes define the SCLK edge on which the MOSI line toggles and the master samples to the MISO line. The modes also define the SCLK steady level: clock/high/low, when the clock is inactive.

Features	Description
	• Each mode is formally defined with a pair of parameters, namely “Clock Polarity (CPOL)” and “Clock Phase” (CPHA). • Refer to Figure 3-139.
Enable or disable transmit and receive modes	• Both DMA and PIO modes are supported on SPIS TX/RX channels.
DMA mode programmable byte length	• The length of TX DMA can be programmable from 1 to 1M Bytes. • The length of RX DMA can be programmable from 1 to (1M-4) Bytes.
RX FIFO full/empty and TX FIFO full/empty interrupts	• RX FIFO full interrupt– RX_FIFO_FULL_EN. SPIS controller sets IRQ status and asserts SPI interrupt to notify the software when RX FIFO is full. The interrupt can be cleared by W1C with RX_FIFO_FULL_CLR and masked by RX_FIFO_FULL_MASK. • RX FIFO empty interrupt – RX_FIFO_EMPTY_EN. SPIS controller sets IRQ status and asserts SPI interrupt to notify the software when RX FIFO is empty. The interrupt can be cleared by W1C with RX_FIFO_EMPTY_CLR and masked by RX_FIFO_EMPTY_MASK. • TX FIFO full interrupt – TX_FIFO_FULL_EN. SPIS controller sets IRQ status and asserts SPI interrupt to notify the software when TX FIFO is full. The interrupt can be cleared by W1C with TX_FIFO_FULL_CLR and masked by TX_FIFO_FULL_MASK. • TX FIFO empty interrupt – TX_FIFO_EMPTY_EN. SPIS controller sets IRQ status and asserts SPI interrupt to notify the software when TX FIFO is empty. The interrupt can be cleared by W1C with TX_FIFO_EMPTY_CLR and masked by TX_FIFO_EMPTY_MASK.
TX FIFO and RX FIFO depths	• 32 × 4 bytes

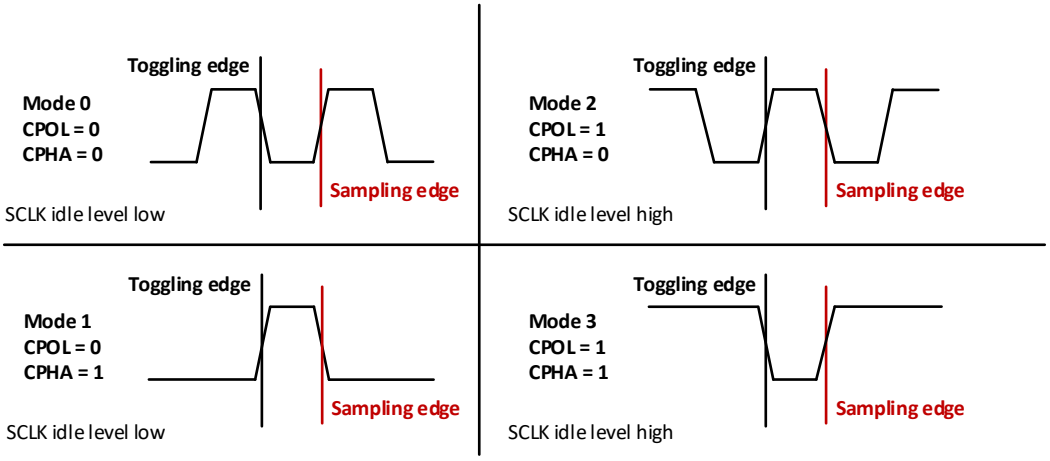


Figure 3-139 Waveforms for Four Communication Modes in SPI

3.12.4.2.6 SPIS Theory of Operations

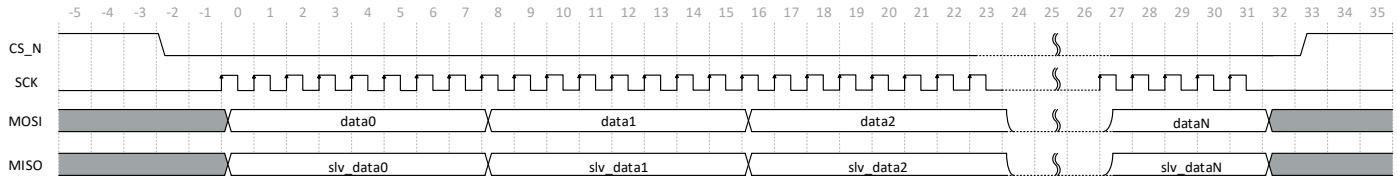


Figure 3-140 SPI Transaction Format

- In DMA mode, data to be transferred should be prepared in advance in SYSRAM.
- In PIO mode, system should push data to be transferred into SPI TX FIFO first. After receiving the START condition, SPIS sends data to the master continuously and at the same time receives data from the master if SPI_CLK and SPI_CS have been activated by the master.

3.12.4.2.7 SPI Slave Timing Characteristics

Table 3-129 and Figure 3-141 present timing characteristics for the SPIS in the device.

Table 3-129 SPIS Timing Characteristics

No.	Parameter		Min	Typ	Max	Unit
	f_{OP_MCK} (2)	SPI clock frequency			26	MHz
SPI02	t_c	Cycle time, SPI clock (SPI_CLK)	38.46 (1)			ns
SPI05	$t_{w\_CLK\_L}$	Pulse duration, SPI_CLK low	14.5			ns
SPI06	$t_{w\_CLK\_H}$	Pulse duration, SPI_CLK high	14.5			ns
SPI07	t_{su_cs}	SPI_CSB falling to SPI_SCK rising setup time	19.2			ns
SPI08	t_{h_cs}	SPI_SCK falling to SPI_CSB rising hold time	19.2			ns
SPI09	t_{su_MOSI}	SPI_MO to SPI_CK rising setup time	7.8			ns
SPI10	t_{h_MOSI}	SPI_SCK rising to SPI_MO hold time	7.8			ns
SPI11	t_{su_MISO}	SPI_MI to _SPI SCK rising setup time requirement	6.6			ns
SPI12	t_{h_MISO}	SPI_SCK rising to SPI_MI hold time requirement	26.4			ns

(1) For maximum operating clock frequency, refer to Table 6-1.

(2) If the SPIM can adjust sample clock delay, the maximum value of f_{OP_MCK} can be up to 52 MHz.

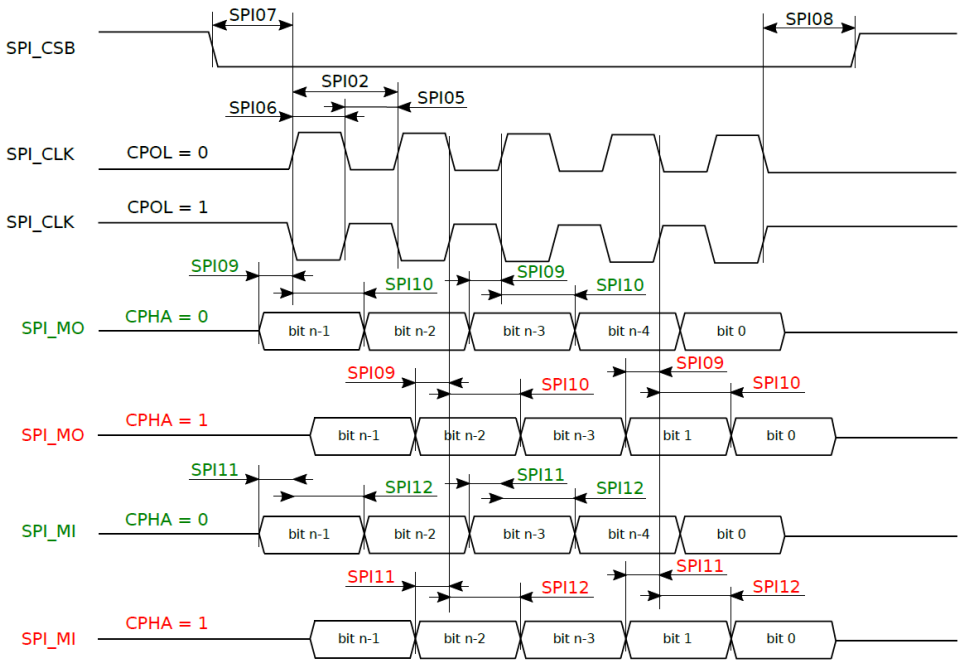


Figure 3-141 SPIS Timing Diagram

3.12.4.2.8 SPIS Programming Guide

3.12.4.2.8.1 PIO Mode

1. Set basic configurations, including SPIS_CFG, SPIS_IRQ_EN, SPIS_FIFO_THR, SPIS_TX_DATA.
2. Write TX memory.
3. Set SPIS_IRQ_MASK.
4. Wait for SPIS_IRQ_ST.
5. Read RX memory.
6. Clear IRQ register.

3.12.4.2.8.2 DMA Mode

1. Set basic configurations, including SPIS_CFG, SPIS_IRQ_EN, SPIS_FIFO_THR, SPIS_TX_SRC, SPIS_RX_DST.
2. Set SPIS DMA configurations, including SPIS_DMA_TRIG_EN, TX_DMA_EM, RX_DMA_EN.
3. Load DMA address.
4. Set SPIS_IRQ_MASK (DMA_DONE_MASK=1, DATA_DONE_MASK=1).
5. Write TX memory.
6. Read RX memory.
7. Wait for SPIS_IRQ_ST.
8. Clear IRQ register.

3.12.5 SuperSpeed Universal Serial Bus (SSUSB)

3.12.5.1 USB MAC Overview

The MediaTek SuperSpeed Universal Serial Bus (SSUSB) module provides four subsystems with integrated PHYs—two SS USB 3.1 Gen1 and two USB 2.0 Dual-Role-Devices (DRD).

- USB Port 0 supports SSUSB 3.1 Gen1 DRD.
- USB Port 1 supports SSUSB 3.1 Gen1 host. The SSUSB Port 1 is shared with PCIe Port 1.
- USB Port 2 and USB Port 3 are USB 2.0 DRD ports.

The module contains a pair of U3 Gen1 PHY and U3 Gen1 MAC for SuperSpeed connection, as well as a pair of U2 PHY and U2 MAC for High-, Full- and LowSpeed connection.

The USB2.0 dual role capability allows the port to support On-The-Go (OTG) host and peripheral functions. When operating in the USB2.0 host role, the port is controlled by the host controller (xHC), which manages all devices connected through its root hub ports. Conversely, when operating in the peripheral role, the port is controlled by the device (DEV) controller.

3.12.5.2 Features

- USB 3.1 SS Gen1 with 5 Gbps TX and 5 Gbps RX (USB Port 0 and USB Port 1 only)
  - Embedded USB 3.1 Gen1 PHY with 32-bit @ 125 MHz PIPE interface
  - U0/U1/U2/U3 states
- USB 2.0 Full-Speed (FS) 12 Mbps and High-Speed (HS) 480 Mbps
  - Embedded USB 2.0 PHY with 16-bit @ 30 MHz UTMI+ interface
  - Lower Power Management (LPM)
- Host role features:
  - Host controller based on eXtensible Host Controller Interface (xHCI) Revision 1.1
  - Dedicated DMA channel for USB 3.1 data transfer
  - Support of all USB compliant data transfer types (Control/Bulk/Interrupt/Isochronous)
  - Support of connection to USB 2.0/USB 3.0 Hubs
  - Support of up to 15 devices
  - Support of up to 64 endpoints
- Device (peripheral) role features:
  - Proprietary application layer device controller
  - Embedded queue management function with scatter/gather DMA capability
  - Shared endpoint and buffer hardware for USB 2.0 and USB 3.1 Gen1 ports
  - Up to 8 OUT endpoints and 8 IN endpoints
  - Up to 8 packet slots for each endpoint
  - Software configurable FIFO size allocation for each endpoint
  - Software configurable transfer type (Bulk/Interrupt/Isochronous) for each endpoint
  - Software configurable interrupt with the following interrupt statuses: VBUS On/Off, suspend/resume, USB Reset
  - Software configurable period from VBUS connection to D+ pull-up
  - Data alignment for Device DMA descriptor: 16-byte
  - Data alignment for Device data: Byte alignment

3.12.5.3 Block Diagram

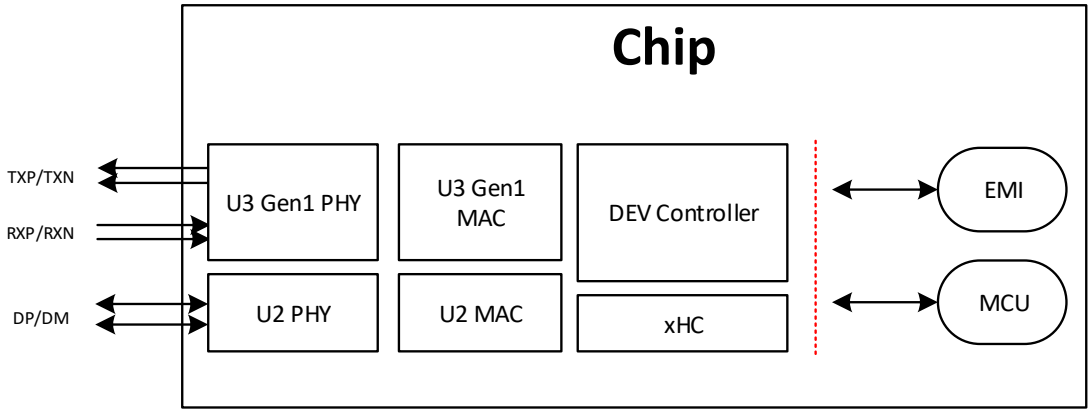


Figure 3-142 Block Diagram of SSUSB

3.12.5.4 USB Signal Descriptions

Table 3-130 presents USB signal descriptions.

Table 3-130 USB Signal Descriptions

Signal Name	Type	Description	Ball Location
USB Port 0			
SSUSB_RXN	AI	USB SuperSpeed receive data negative	AK31
SSUSB_RXP	AI	USB SuperSpeed receive data positive	AJ31
SSUSB_TXN	AO	USB SuperSpeed transmit data negative	AH32
SSUSB_TXP	AO	USB SuperSpeed transmit data positive	AH33
USB_DP_P0	AIO	USB D+ bi-directional differential data	AM35
USB_DM_P0	AIO	USB D- bi-directional differential data	AM36
IDDIG	DI	USB OTG ID. Cable end detector: GND: micro-A Floating: micro-B	AL9
USB_DRVVBUS	DO	USB drive VBUS—signal to external power switch enable	AM10
VBUSVALID	DI	Digital VBUS—valid signal from external circuitry	AU18
USB Port 1			
PCIE_LN0_RXP_P1	AI	USB SuperSpeed receive data negative(1)	V33
PCIE_LN0_RXN_P1	AI	USB SuperSpeed receive data positive(1)	V34
PCIE_LN0_TXP_P1	AO	USB SuperSpeed transmit data negative(1)	W31
PCIE_LN0_TXN_P1	AO	USB SuperSpeed transmit data positive(1)	W32
USB_DP_P1	AIO	USB D+ bi-directional differential data	AC31
USB_DM_P1	AIO	USB D- bi-directional differential data	AC32
VBUSVALID_1P	DI	Normally not used	AT18
USB Port 2			
USB_DP_P2	AIO	USB D+ bi-directional differential data	AD8
USB_DM_P2	AIO	USB D- bi-directional differential data	AE8
IDDIG_1P	DI	USB OTG ID. Micro-A/B cable end detector	H31, AL7
USB_DRVVBUS_1P	DO	USB drive VBUS—signal to external power switch enable	G34, AL8

Signal Name	Type	Description	Ball Location
VBUSVALID_2P	DI	Digital VBUS-valid signal from external circuitry	AR8
USB Port 3			
USB_DP_P3	AIO	USB D+ bi-directional differential data	AH7
USB_DM_P3	AIO	USB D- bi-directional differential data	AG7
IDDIG_2P	DI	USB OTG ID. Micro-A/B cable end detector	G35, AT16
USB_DRVVBUS_2P	DO	USB drive VBUS—signal to external power switch enable	H33, AN15
VBUSVALID_3P	DI	Digital VBUS-valid signal from external circuitry	AP8

(1) Shared with PCIe Port 1.

3.12.5.5 Function Description

3.12.5.5.1 Host Architecture

The SSUSB is the designated application for the MediaTek USB host controllers as mentioned in Section 3.12.5.1. Each PHY is equipped with its own MAC for protocol packet management. Figure 3-143 illustrates the architecture of the SSUSB host.

The xHCI controller manages all endpoint and device resources. Dynamic allocation of resources for different ports is achievable by software, which can also enable or disable each port separately.

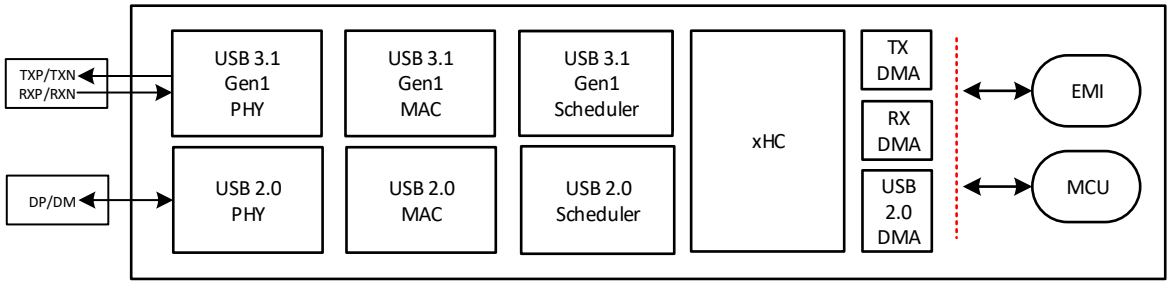


Figure 3-143 SSUSB host architecture

3.12.5.5.1.1 Features

- Hardware supports USB 3.1 SuperSpeed Gen1 with 5 Gb/s TX and 5 Gb/s RX bandwidth.
- Hardware supports USB 2.0 with Full-speed 12 Mbps/High-speed 480 Mbps.
- Embedded USB 3.1 Gen1PHY with 32-bit 125 MHz PIPE interface
- Embedded USB 2.0 PHY with 16-bit/30 MHz UTMI
- AHB interface for register access
- AXI3 interface for DMA access
- Extensible xHCI Revision 1.1 based host controller
- Lower Power Management (LPM) on USB 2.0 port
- U0/U1/U2/U3 state on USB 3.1 Gen1 port
- Dedicated DMA channel for USB 3.1 data transfer
- Supports all USB compliant data transfer types with control/bulk/interrupt/isochronous transfer
- Compatible to connect to USB 2.0/USB3.0 hubs
- Maximum 15 devices
- Maximum 32 endpoints

- Bulk stream: Not supported
- Transaction Translator (TT) (Multiple Transaction Translator (MTT) not supported)

3.12.5.5.2 Device Architecture Overview

Figure 3-144 illustrates the architecture of the SSUSB device function. The USB 2.0 and USB 3.1 Gen1 PHY and MAC operate independently for different transmission lines. As only one link can connect to the USB host at a time, they share the same endpoint and buffer management unit.

The linked-list queue of the SSUSB device is inherited from the MediaTek unified USB with similar descriptor definition. On a SuperSpeed link, it works on the TX and RX concurrently.

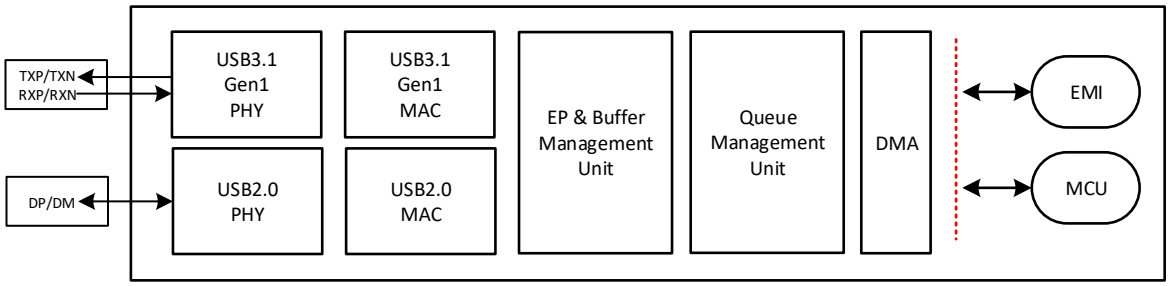


Figure 3-144 SSUSB Device Architecture

3.12.5.5.2.1 Features

- Hardware supports USB 3.1 Gen1 and USB 2.0 High-speed/Full-speed
- Embedded USB 3.1 PHY with 32-bit/125 MHz (Gen1) PIPE interface
- Embedded USB 2.0 PHY with 16-bit/30 MHz UTMI
- AHB interface for register access
- AXI3 interface for DMA access
- Embedded queue management function with scatter/gather DMA capability
- Proprietary application layer device controller with linked list queue and scatter/gather DMA
- Lower Power Management (LPM) on USB 2.0 port
- U0/U1/U2/U3 state on USB 3.1 Gen1 port
- Shared endpoint and buffer hardware for USB 2.0 and USB 3.1 Gen1 port
- Hardware configurable to USB 2.0 only device
- Hardware configurable up to 8 OUT endpoints and 8 IN endpoints
- Hardware configurable up to 8 packet slots for each endpoint separately
- Software configurable FIFO size allocation for each endpoint separately
- Software configurable transfer type to Bulk/Interrupt/Isochronous for each endpoint
- Software configurable Interrupt with the following interrupt statuses: VBUS On/Off, suspend/resume, USB Reset
- Software configurable for the period from VBUS connection to D+ pull up
- Data alignment for Device DMA Descriptor “16-byte alignment” in SDRAM
- Data alignment for Device Data: “Byte alignment”
- Bulk stream: Not supported

3.12.5.6 Theory of Operations

3.12.5.6.1 USB Host Initialization

- After Power-On-Reset (POR), USB powers down by default to save power.
- Software must execute the following steps before using USB. Refer to Table 3-131 for details.
  For the MediaTek scheduling algorithm initialization, refer to the kernel drivers implemented in *xhci_mtk_sch_init ()* of */usb/host/mtk-xhci-sch.c* file.

Table 3-131 USB Host Initialization

Step	Address	Register Name	Local Address	R/W	Value	Description
USB Initialization						
1	ssusb_sifslv_ippc_Base address + 0x00000	SSUSB_IP_PW_CTRL0	SSUSB_IP_SW_RST [0]	W	1'b0	USB Software Reset When this bit is set, the whole USB is reset. Write “0” to release reset. Setting flow: SSUSB_IP_SW_RST [0] = “1”, delay 1 µs. SSUSB_IP_SW_RST [0] = “0”.
2	ssusb_sifslv_ippc_Base address + 0x0004	SSUSB_IP_PW_CTRL1	SSUSB_IP_HOST_PDN [0]	W	1'b0	This bit is USB power-down bit. Write “0” to disable USB power-down.
Enable USB 3.1 Gen1 Port						
3	ssusb_sifslv_ippc_Base address + 0x0030	SSUSB_U3_CTRL_0P	SSUSB_U3_PORT_DIS [0]	W	1'b0	USB 3.0 port0 disable bit. “0”: USB 3.0 port0 is enabled.
4	ssusb_sifslv_ippc_Base address + 0x0030	SSUSB_U3_CTRL_0P	SSUSB_U3_PORT_PDN [1]	W	1'b0	USB 3.0 port0 power-down bit. “0”: USB 3.0 port0 is powered on.
5	ssusb_sifslv_ippc_Base address + 0x0030	SSUSB_U3_CTRL_0P	SSUSB_U3_PORT_HOST_SEL [2]	W	1'b1	This bit is Host mode selection of USB 3.0 port0. “1”: This port is selected for Host mode.
Enable USB 2.0 Port						
6	ssusb_sifslv_ippc_Base address + 0x0050	SSUSB_U2_CTRL_0P	SSUSB_U2_PORT_DIS [0]	W	1'b0	USB 2.0 port0 disable bit. “0”: USB 2.0 port0 is enabled.
7	ssusb_sifslv_ippc_Base address + 0x0050	SSUSB_U2_CTRL_0P	SSUSB_U2_PORT_PDN [1]	W	1'b0	USB 2.0 port0 power-down bit. “0”: USB 2.0 port0 is powered on.
8	ssusb_sifslv_ippc_Base address + 0x0050	SSUSB_U2_CTRL_0P	SSUSB_U2_PORT_HOST_SEL [2]	W	1'b1	This bit is Host mode selection of USB 2.0 port0. “1”: This port is selected for Host mode.
Check reference clock stability before software proceeds (i.e., Software proceeds after the following status bits are asserted)						
9	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_SYSPLL_STABLE [0]	R		When this bit is 1'b1, SYSPLL for USB is stable.
10	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_REF_RST_B_STS [8]	R		When this bit is 1'b1, it means that reset for reference clock (ref_ck) domain is inactive.

Step	Address	Register Name	Local Address	R/W	Value	Description
11	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_SYS125_RST_B_STS [10]	R		When this bit is 1'b1, it means that reset for sys125_ck domain is inactive.
12	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_XHCI_RST_B_STS [11]	R		When this bit is 1'b1, it means that reset for xhci_ck domain is inactive.

3.12.5.6.2 USB Device Initialization

- After Power-On-Reset, USB powers down by default to save power.
- Software must perform the following steps before using USB. Refer to Table 3-132 for details.

Table 3-132 USB Device Initialization

Step	Address	Register Name	Local Address	R/W	Value	Description
USB Initialization						
1	ssusb_sifslv_ippc_Base address + 0x0000	SSUSB_IP_PW_CTRL0	SSUSB_IP_SW_RST [0]	W	1'b0	USB Software Reset. When this bit is set, the whole USB is reset. Write “0” to release reset.
2	ssusb_sifslv_ippc_Base address + 0x0008	SSUSB_IP_PW_CTRL2	SSUSB_IP_DEV_PDN[0]	W	1'b0	SSUSB IP Dev Power Down. When this bit is set, the whole CKBG can be powered down. Write “0” to release reset.
Enable USB 3.1 Gen1 Port (If device is USB 2.0 only, please skip it.)						
3	ssusb_sifslv_ippc_Base address + 0x0030	SSUSB_U3_CTRL_0P	SSUSB_U3_PORT_DIS [0]	W	1'b0	USB 3.0 port0 disable bit “0”: USB 3.0 port0 is enabled.
4	ssusb_sifslv_ippc_Base address + 0x0030	SSUSB_U3_CTRL_0P	SSUSB_U3_PORT_PDN [1]	W	1'b0	USB 3.0 port0 power-down bit “0”: USB 3.0 port0 is powered on.
5	ssusb_sifslv_ippc_Base address + 0x0030	SSUSB_U3_CTRL_0P	SSUSB_U3_PORT_HOST_SEL [2]	W	1'b0	This bit is Host mode selection of USB 3.0 port0. “0”: This port is selected for Device mode.
Enable USB 2.0 Port						
6	ssusb_sifslv_ippc_Base address + 0x0050	SSUSB_U2_CTRL_0P	SSUSB_U2_PORT_DIS [0]	W	1'b0	USB 2.0 port0 disable bit “0”: USB 2.0 port0 is enabled.
7	ssusb_sifslv_ippc_Base address + 0x0050	SSUSB_U2_CTRL_0P	SSUSB_U2_PORT_PDN [1]	W	1'b0	USB 2.0 port0 power-down bit “0”: USB 2.0 port0 is powered on.
8	ssusb_sifslv_ippc_Base address + 0x0050	SSUSB_U2_CTRL_0P	SSUSB_U2_PORT_HOST_SEL [2]	W	1'b0	This bit is Host mode selection of USB 2.0 port0 “0”: This port is selected for Device mode.
Check reference clock stability before software proceeds (i.e., Software proceeds after the following status bits are asserted)						
9	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_SYS125_RST_B_STS [10]	R		When this bit is 1'b1, it means that reset for sys125_ck domain is inactive.

Step	Address	Register Name	Local Address	R/W	Value	Description
10	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_U3_MAC_RST_B_STS[16]	R		When this bit is 1'b1, it means that reset for mac3_mac_ck domain is inactive (If device is USB 2.0 only, please skip it.).
11	ssusb_sifslv_ippc_Base address + 0x0014	SSUSB_IP_PW_STS2	SSUSB_U2_MAC_SYS_RST_B_STS [0]	R		When this bit is 1'b1, it means that reset for mac2_sys_ck domain is inactive.
Turn on the USB connection by speed. (USB 3.1)						
12	ssusb_sifslv_ippc_Base address + 0x00c8	SSUSB_IP_SPARE0	SSUSB_U3_PORT_SS_SUP_SPEED[1:0]	W	User defined	Users can set the register to force operation speed.
13	ssusb_usb3_mac_csr_Base address + 0x001c	USB3_CONFIG	USB3_EN[0]	W	1'b1	Enable SuperSpeed function.
Turn on the USB connection by speed. (USB 2.0)						
14	ssusb_usb3_mac_csr_Base address + 0x001c	USB3_CONFIG	USB3_EN[0]	W	1'b0	Disable SuperSpeed function.
15	ssusb_usb2_csr_Base address + 0x0004	POWER_MANAGEMENT	HS_ENABLE[5]	W	1'b1	Enable high speed function.
16	ssusb_usb2_csr_Base address + 0x0004	POWER_MANAGEMENT	SUSPENDM_ENABLE[0]	W	1'b1	Enable the SUSPENDM output.
17	ssusb_usb2_csr_Base address + 0x0004	POWER_MANAGEMENT	soft_conn[6]	W	1'b1	Enable D+/D- lines.

Note: On USB 3.1 Gen1/USB 2.0 port power down

- Registers in SSUSB_USB3_SYS/SSUSB_USB3_MAC cannot be read correctly when SSUSB_U3_PORT_PDN or SSUSB_U3_PORT_DIS = 1.
- Registers in SSUSB_USB2 cannot be read correctly when SSUSB_U2_PORT_PDN or SSUSB_U2_PORT_DIS = 1.
- After setting SSUSB_U*_PORT_PDN and SSUSB_U*_PORT_DIS to 0, wait until SYSPLL becomes stable (SSUSB_IPCTL .U3D_SSUSB_IP_PW_STS1. SSUSB_SYSPLL_STABLE).

3.12.5.6.3 USB Endpoint Initialization

3.12.5.6.3.1 EP0 Initialization

- EP0 hardware should be configured properly before a successful USB enumeration.
- EP0 is controlled by USB only and software fills the correct value into EP0 CSR. The EP0 initialization is included in the USB initial flow. And it is not recommended to modify it except for the maximum packet size.

Table 3-133 EP0 Programming Sequence

Step	Address	Register Name	Local Address	R/W	Value	Description
1	ssusb_dev_Base address + 0x0100	EP0CSR	EP0_MAXPKTSZ0[9:0]	W	USER DEFINED	Set maximum packet size For example, 10'd64
2	ssusb_dev_Base address + 0x0088	EPIESR	EP0IESR[0]	W	1'b1	Endpoint 0 interrupt enable setting
3	ssusb_dev_Base address + 0x0088	EPIESR	SETUPENDIESR[16]	W	1'b1	Endpoint 0 Setup End interrupt enable setting

3.12.5.6.3.2 EPn Initialization

The suggested programming sequence of TX EPn (e.g., EP1) is shown in the table below.

Table 3-134 TX EPn Programming Sequence

Step	Address	Register Name	Local Address	R/W	Value	Description
1	ssusb_dev_Base address + 0x0110	TX1CSR0	TX_TXMAXPKTSZ[10:0]	W	USER DEFINED	Set maximum packet size. For example, 11'd64.
2	ssusb_dev_Base address + 0x0114	TX1CSR1	SS_TX_BURST[3:0]	W	USER DEFINED	Set burst size.
3	ssusb_dev_Base address + 0x0114	TX1CSR1	TX_MULT[23:21]	W	USER DEFINED	Set TX_MULT size.
4	ssusb_dev_Base address + 0x0114	TX1CSR1	TX_MAX_PKT[30:24]	W	USER DEFINED	Number of packets = (SS_TX_BURST+ 1) x (TX_MULT + 1) -1 (Isochronous endpoint only).
5	ssusb_dev_Base address + 0x0118	TX1CSR2	TXFIFOADDR[12:0]	W	USER DEFINED	Start address of the selected TX endpoint FIFO
6	ssusb_dev_Base address + 0x0114	TX1CSR1	TX_SLOT[13:8]	W	USER DEFINED	Set slot number of hardware.
7	ssusb_dev_Base address + 0x0118	TX1CSR2	TXFIFOSEGSIZE[19:16]	W	USER DEFINED	Set FIFO segment size of hardware layout. Indicate the TX FIFO size of 2^n bytes.
8	ssusb_dev_Base address + 0x0114	TX1CSR1	TXTYPE[5:4]	W	USER DEFINED	Select the required transfer type for the TX endpoint. 2'b00: Bulk 2'b01: Interrupt 2'b10: Isochronous
9	ssusb_dev_Base address + 0x0118	TX1CSR2	TXBINTERVAL[31:24]	W	USER DEFINED	Interval for servicing the endpoint for data transfer. For Isochronous/Interrupt transfer.
10	ssusb_dev_Base address + 0x0110	TX1CSR0	TX_DMAREQEN[29]	W	1'b1	Enable EP1 DMA request for the TX endpoint.
11	ssusb_dev_Base address + 0x0708	QIESR0	TXQ_DONE_IESR[1]	W	1'b1	Enable EP1 TX QMU Done interrupt.

The suggested programming sequence of RX EPn (e.g.,EP2) is shown in the table below.

Table 3-135 RX EPn Programming Sequence

Step	Address	Register name	Local address	R/W	Value	Description
1	ssusb_dev_Base address + 0x0220	RX2CSR0	RX_RXMAXPKTSZ[10:0]	W	USER DEFINED	Set the maximum packet size. For example, 11'd64
2	ssusb_dev_Base address + 0x0224	RX2CSR1	SS_RX_BURST[3:0]	W	USER DEFINED	Set burst size.
3	ssusb_dev_Base address + 0x0224	RX2CSR1	RX_MULT[23:21]	W	USER DEFINED	Set RX_MULT size.
4	ssusb_dev_Base address + 0x0224	RX2CSR1	RX_MAX_PKT[30:24]	W	USER DEFINED	Number of packets = (SS_RX_BURST+ 1) x

Step	Address	Register name	Local address	R/W	Value	Description
						(RX_MULT + 1) -1 (Isochronous endpoint only).
5	ssusb_dev_Base address + 0x0228	RX2CSR2	RXFIFOADDR[12:0]	W	USER DEFINED	Start address of the selected Rx endpoint FIFO.
6	ssusb_dev_Base address + 0x0224	RX2CSR1	RX_SLOT[13:8]	W	USER DEFINED	Set slot number of hardware.
7	ssusb_dev_Base address + 0x0228	RX2CSR2	RXFIFOSEGSIZE[19:16]	W	USER DEFINED	Set FIFO segment size of hardware layout. Indicate the RX FIFO size of 2^n bytes.
8	ssusb_dev_Base address + 0x0224	RX2CSR1	RX_TYPE[5:4]	W	USER DEFINED	Select the required transfer type for the RX endpoint. 2'b00: Bulk 2'b01: Interrupt 2'b10: Isochronous
9	ssusb_dev_Base address + 0x0228	RX2CSR2	RXBINTERVAL[31:24]	W	USER DEFINED	Interval for servicing the endpoint for data transfer. For Isochronous/Interrupt transfer
10	ssusb_dev_Base address + 0x0220	RX2CSR0	RX_DMAREQEN[29]	W	1'b1	Enable EP2 DMA request for the RX endpoint.
11	ssusb_dev_Base address + 0x0708	QIESR0	TXQ_DONE_IESR [18]	W	1'b1	Enable EP2 QMU Done interrupt.

3.12.5.7 Programming Guide

3.12.5.7.1 USB Host

There are certain differences between the MediaTek Host Controller (xHC) and the standard xHCI.

- The MediaTek xHC does not implement completion codes because some TRB types are defined in xHCI specification. A detailed description is in Section 3.12.5.7.1.1.
- The MediaTek xHC proposes a scheduling mechanism for synchronous endpoint to simplify the hardware design. The mechanism is described in Section 3.12.5.7.1.2.
- Interrupt moderation Interval (IMODI): The standard xHCI interval is 250 ns, while the MediaTek xHC interval is 2 µs.
- Only one of the xHCI extended capability codes, “Supported Protocol” (ID code = 'd2), is implemented in the MediaTek xHC.
- Configuration Information Capability (CIC) feature
- The followings features are not implemented in the MediaTek xHC:
  - Frame Length Adjustment Register (FLADJ)
  - The latency tolerance messaging LTV, LTM, and BEL
  - The BUS_INTERVAL_ADJUSTMENT_MESSAGE notification packet

3.12.5.7.1.1 Unsupported TRB Types and Completion Codes

The following TRB types are **not** implemented in the MediaTek xHC:

- Force event command TRB (TRB Type = 18, optional normative)
- Negotiate bandwidth command TRB (TRB Type = 19, optional normative)
- Set latency tolerance value command TRB (TRB Type = 20, optional normative)
- Get port bandwidth command TRB (TRB Type = 21)
- Bandwidth request event TRB (TRB Type = 35)
- Doorbell event TRB (TRB Type = 36)

The following completion codes are **not** implemented in MediaTek xHC:

- Data buffer error (error value = 2)
- Bandwidth error (error value = 8)
- VF event ring full error (error value = 16)
- Bandwidth overrun error (error value = 18)
- Incompatible device error (error value = 22)
- Max exit latency too large error (error value = 29)
- Event lost error (error value = 32)
- Undefined error (error value = 33)
- Secondary bandwidth error (error value = 35)
- Split transaction error (error value = 36)
- Vendor defined error (error value = 192 to 223)
- Vendor defined info (error value = 224 to 255)

3.12.5.7.1.2 Scheduling of Synchronous Endpoint

To simplify the hardware design for bandwidth calculation and scheduling on synchronous endpoint, a proprietary scheduling algorithm is proposed. To implement this algorithm for the MediaTek Host Controller Driver (xHCD), it is necessary to path the standard Linux xHCD driver. The patch includes the following two steps:

1. Calculate whether there is enough bandwidth reserved for the endpoint(s) to be added.
2. Determine a set of parameters specifying the scheduling for synchronous endpoint(s) to be added.

- **Bandwidth Calculation**

Due to offloading of bandwidth calculation by the xHCD, the MediaTek xHC is able to process the following xHCI commands with greater ease:

1. Reset the device
2. Configure the endpoint (with DC = 1 of Drop Flag(s) = 1)
3. Disable the slot command

In addition, the get port bandwidth command TRB is never replaced on the command ring for the MediaTek xHC because all bandwidth information is directly visible to the xHCD.

- **Decide the Software Scheduling Parameters**

The xHCD implements the proprietary scheduling algorithm to enable easy scheduling of synchronous endpoints by the xHC. Prior to issuing the configure endpoint TRB, a set of parameters is defined for each device slot and its related

synchronous endpoint. To incorporate these proprietary parameters into the MediaTek xHC, certain reserved fields of the endpoint context are utilized to store their values.

The extra software scheduling parameters are implemented through the commands specified in the xHCI specification, as some reserved DWs in the endpoint context are still available. These additional fields are exclusively reserved for synchronous endpoint, such as isochronous and interrupt endpoints. Table 3-136 shows the modified endpoint context, with the extra defined fields highlighted in orange, and Table 3-137 lists the definitions.

Table 3-136 Endpoint Context

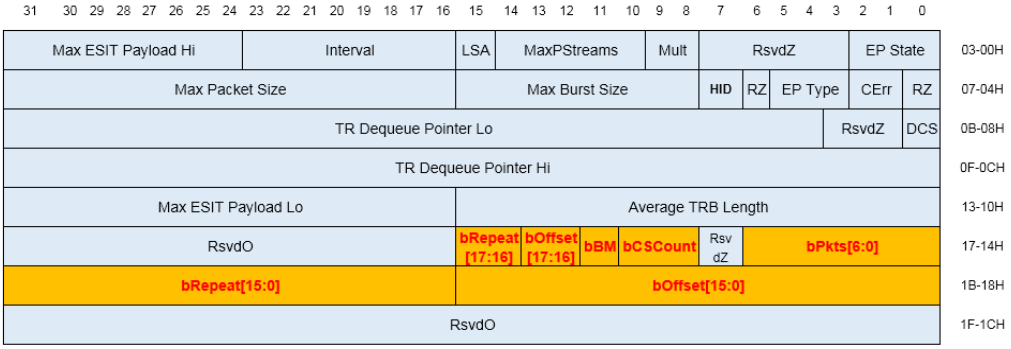


Table 3-137 Extra Defined Fields for Endpoint Context

Field Name	Location	Definition
bPkts[6:0]	DW5[6:0]	Number of packets to be transferred in the scheduled microframes (Unit: microframe 125 µs)
bCSCount[2:0]	DW5[10:8]	• Host will trigger the number of Complete Spilt (CS). • This field is only for Split Transaction at Full-speed/Low-speed. • For full-speed/low-speed isochronous IN and interrupt EPs, this represents the pre-defined number of CS to be in a service interval. • For full-speed/low-speed isochronous OUT EPs, this represents the pre-defined number of SS (Start Spilt) to be in a service interval.
bBM (bBurstmode)	DW5[11]	• Burst mode for scheduling • Normal burst mode is used by default. • 0: Normal burst mode. Distribute the bMaxBurst+1 packets for a single burst according to bPkts and bRepeat, and repeat the burst multiple times. • 1: Distribute the (bMaxBurst+1)*(Mult+1) packets according to bPpkts and bRepeat.
bOffset[17:0]	DW6[15:0] DW5[13:12]	Which microframe of the interval that is transferred should be scheduled at the first time within the interval (Unit: microframe 125 µs)
bRepeat[17:0]	DW6[31:16] DW5[15:14]	The time gap between two microframes, in which transfer USB packets are scheduled within an interval (Unit: microframe 125 µs)

The relation between the extra defined field parameters and microframes is illustrated in the figure below.
For example: EP5 parameters, mult = 0, maxburst = 6, interval = 4

1. EP5 service interval = 2^(4 - 1) = 8 microfames. EP5 total pkts = (mult + 1) x (maxburst + 1) = 7
2. The parameters of the extra defined fields can be scheduled as the following figure shows.

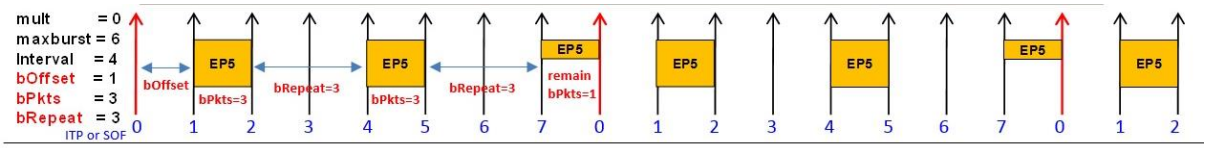


Figure 3-145 Relation between Extra Defined Fields Parameters and Microframes

The relation between the extra defined field parameters and microframes for split transaction at full-speed/low-speed is illustrated in the figure below.

For example: Split IN Transaction for FS/LS. If EP internal = 1 (1ms)

1. Service Interval = 8 microframes.
2. The parameter of the extra defined fields bCScount value is as shown below.

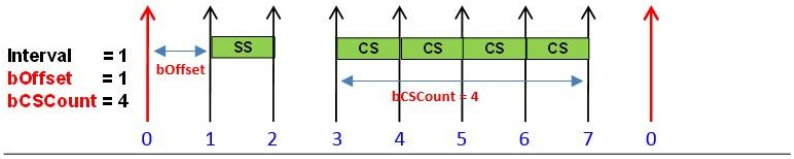


Figure 3-146 Relation between bCSCount Parameters and Microframes

The software flow of “xhci_add_endpoint” configuration by a standard xHCI driver is illustrated in Figure 3-147. To add extra defined field parameters to endpoint context, a sub-flow is patched. This patched sub-flow is marked by red dash line in Figure 3-148.

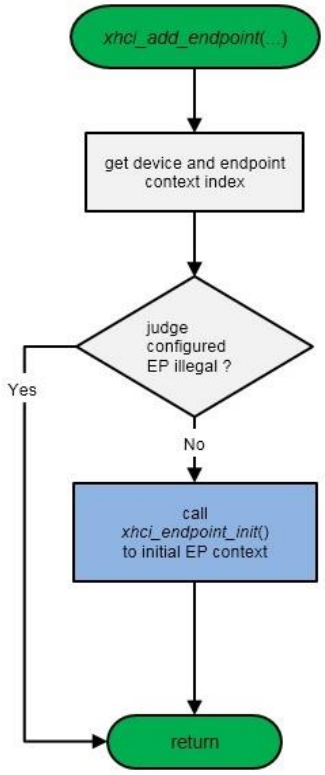


Figure 3-147 Standard xHCI_add_endpoint() Flow

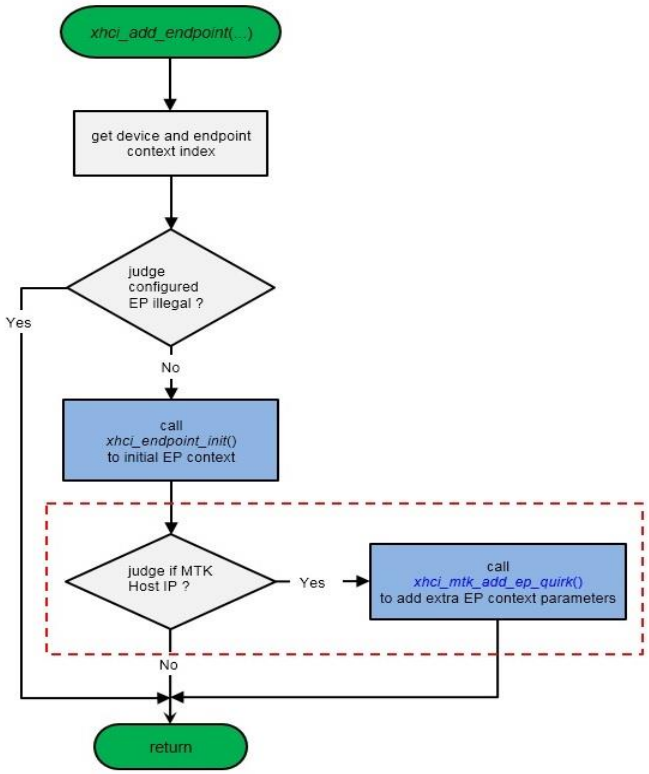


Figure 3-148 Patched xHCI_add_endpoint() Flow

The software flow of "xhci_drop_endpoint" configured by a standard xHCI driver is illustrated in Figure 3-149. To drop software recording extra defined field parameters, a sub-flow is patched. This patched sub-flow is marked by red dash line in Figure 3-150.

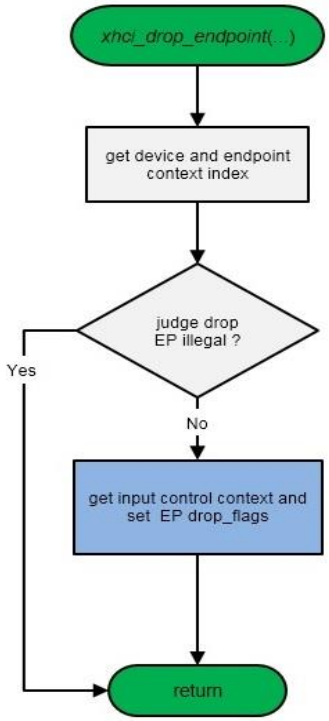


Figure 3-149 Standard xHCI_drop_endpoint() Flow

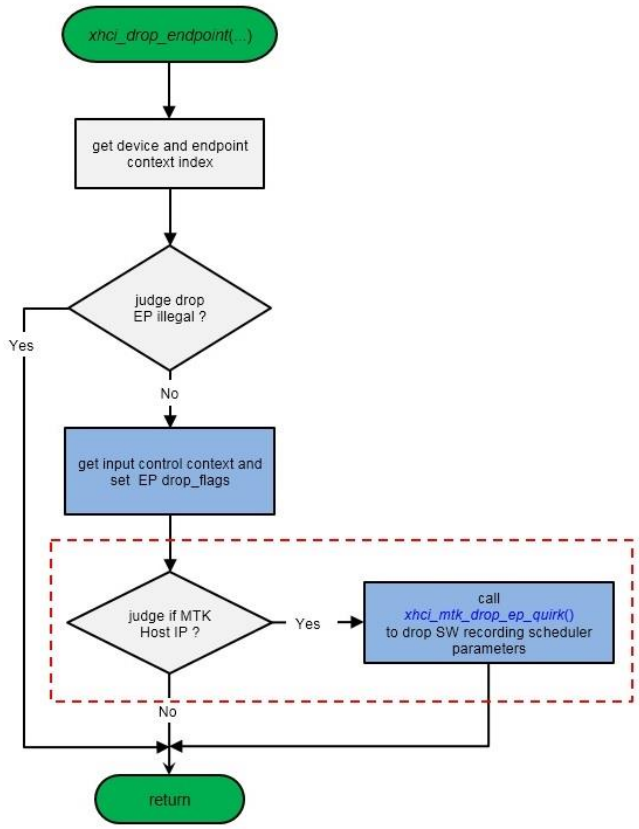


Figure 3-150 Patched xHCI_drop_endpoint() Flow

Refer to the Linux standard xHCI kernel driver (version 4.5 or later) in *drivers/usb/host/mtk-xhci-sch.c* file for more details about the extra defined field parameters for endpoint scheduling.

3.12.5.7.2 USB Device

3.12.5.7.3 EP0 Top Programming Outline

The EP0 control is a state machine with three state modes:

- Idle mode
- TX mode
- RX mode

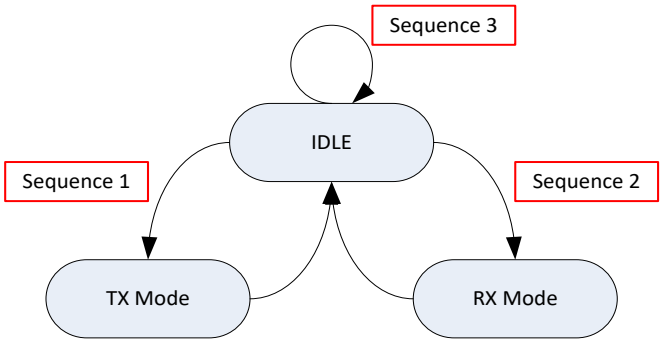


Figure 3-151 EP0 States

- EP0 interrupt is generated when

- EP0CSR.RxPktRdy bit is set after data packet has been received and stored into FIFO0;
- Data packet in FIFO0 has been sent to host successfully;
- EP0CSR.SentStall bit is set after the host receives STALL;
- EPISR.SETUPENDISR bit is set after SETUP transaction is received in the DATA/STATUS phase.

- *EPISR.SETUPENDISR* indicates that the current control transfer is aborted.
  - Set when
    - IN transaction is received after *EP0CSR.DATAEND* is set to leave the TX mode (STATUS phase).
    - OUT transaction is received after *EP0CSR.DATAEND* is set to leave the RX mode (STATUS phase).
    - SETUP transaction is received in the TX mode or RX mode (DATA phase).
  - Cleared when
    - Software sets EPISR.SETUPENDISR.
    - Software can unload FIFO and decode the new command.
- EP0CSR.DataEnd indicates that data phase has finished.
  - Set by software when
    - In the TX mode or RX mode, the amount of data required by host is sent/received.
    - In the TX mode, a smaller amount of data required by the host is sent. The device has to notify the host by sending a short packet.
    - In the RX mode, a short packet is received.
  - Cleared by hardware when
    - (Normal) Status phase finishes successfully.
    - (Error) EPISR.SETUPENDISR is set.

3.12.5.7.3.1 Idle Mode

Upon power-on or reset, EP0 goes into the “IDLE” mode. After receiving a SETUP transaction:

- *EP0CSR.SetupPktRdy* is set.
- *EP0CSR.DPHTX* is cleared.
- An interrupt is generated to notify software.

The software unloads FIFO and decodes the command which, depending on the type, enables the software to do the following:

- *(Sequence 1/IN DATA) W1C EP0CSR.SetupPktRdy* and set *EP0CSR.DPHTX*.
  EP0 goes to TX mode.
- *(Sequence 2/OUT DATA) W1C EP0CSR.SetupPktRdy*.
  EP0 goes to RX mode.
- *(Sequence 3/NO DATA)* process command. Then, *W1C EP0CSR.SetupPktRdy* and set *EP0CSR.DataEnd* (or set *EP0CSR.SendStall*) simultaneously.
  EP0 stays in “IDLE” state.

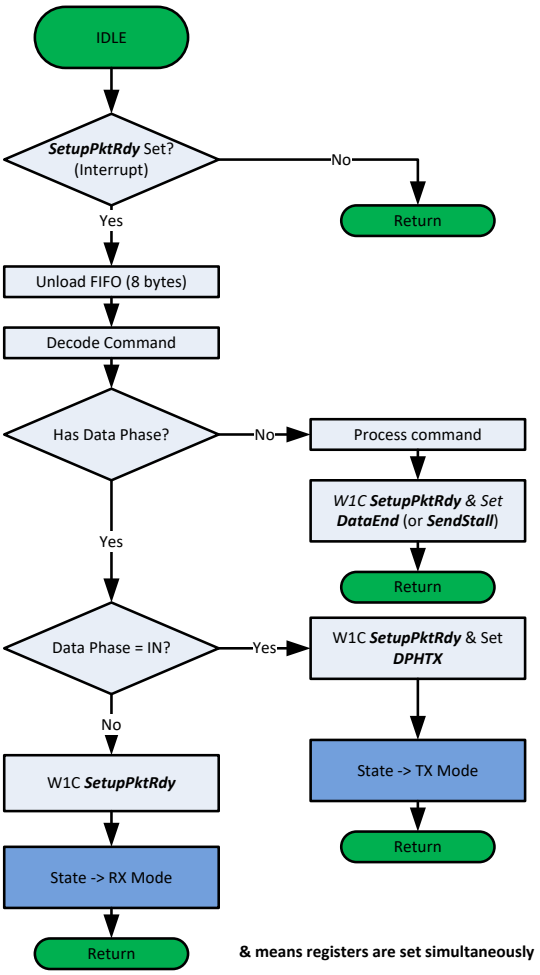


Figure 3-152 EP0 Idle Flow Chart

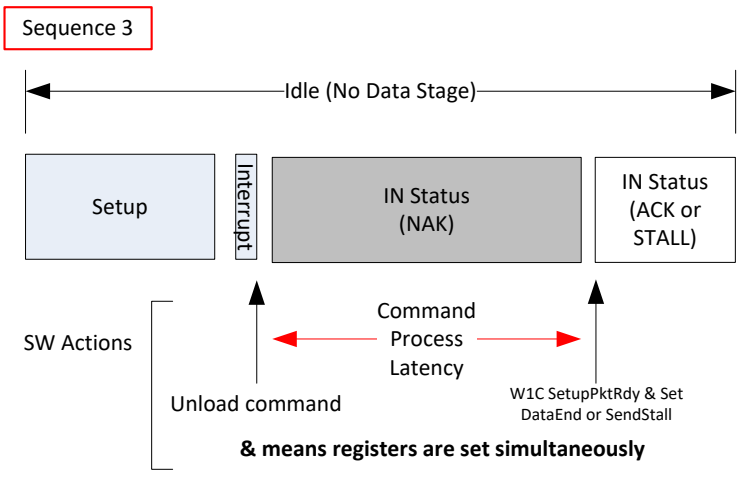


Figure 3-153 EP0 Idle

3.12.5.7.3.2 TX Mode

- **Normal Flow**
  - Software loads data packet (<=*EP0CSR.MaxPktSz0*) to FIFO0, and sets *EP0CSR.TxPktRdy* to send it to the host.

- If *EP0CSR.AutoSet* is set, software only needs to set *EP0CSR.TxPktRdy* for the last data packet (a short packet).
- After sending data packet to the host, EP0CSR.FIFOFull is cleared and an interrupt is generated to notify software.
- Software repeats the previous step until the required amount of data is sent, and set *EP0CSR.DATAEND* to leave DATA phase.

- **Error Cases**
  - If a SETUP transaction is received in TX mode (DATA phase), *EPISR.SETUPENDISR* is set and an interrupt is generated. Software aborts the current command and moves on to decode the new command.

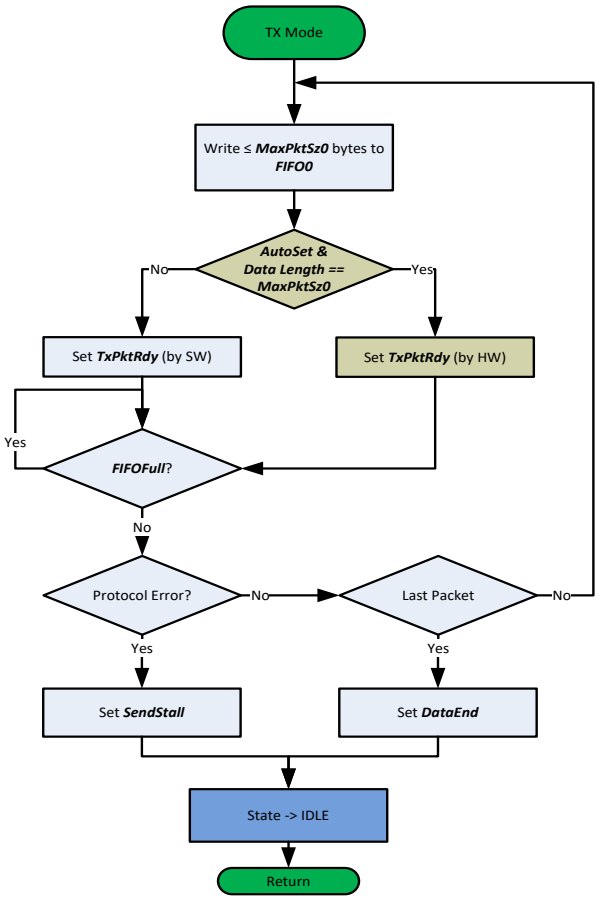


Figure 3-154 EP0 TX Mode Flow Chart

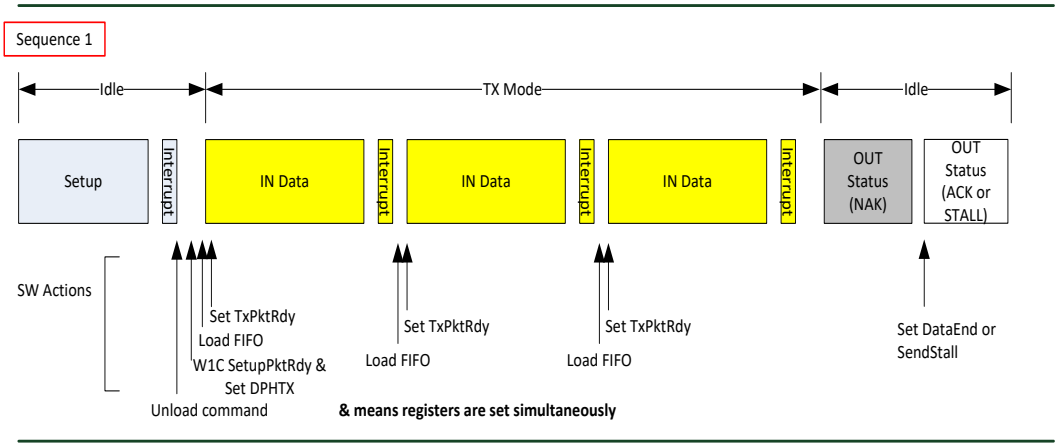


Figure 3-155 EP0 TX Mode Flow Chart

3.12.5.7.3.3 RX Mode

- **Normal flow**
  - After receiving data packet (<=*EP0CSR.MaxPktSz0*), hardware sets *EP0CSR.RxPktRdy* and generates an interrupt.
  - Software unloads data packet (<=*EP0CSR.MaxPktSz0*) from FIFO0 and *W1C EP0CSR.RxPktRdy*.
  - If *EP0CSR.AutoClear* is set, *EP0CSR.RxPktRdy* is cleared automatically after data packet is unloaded, unless the data packet is of a size of 0. Software needs to *W1C EP0CSR.RxPktRdy* under this condition.
  - Software repeats the previous step until the required amount of data is received or a short packet is received, and set *EP0CSR.DATAEND* to leave the DATA phase.

- **Error cases**
  - If a SETUP transaction is received in the RX mode (DATA phase), *EPISR.SETUPENDISR* is set and an interrupt is generated. Software aborts the current command and moves on to decode the new command.

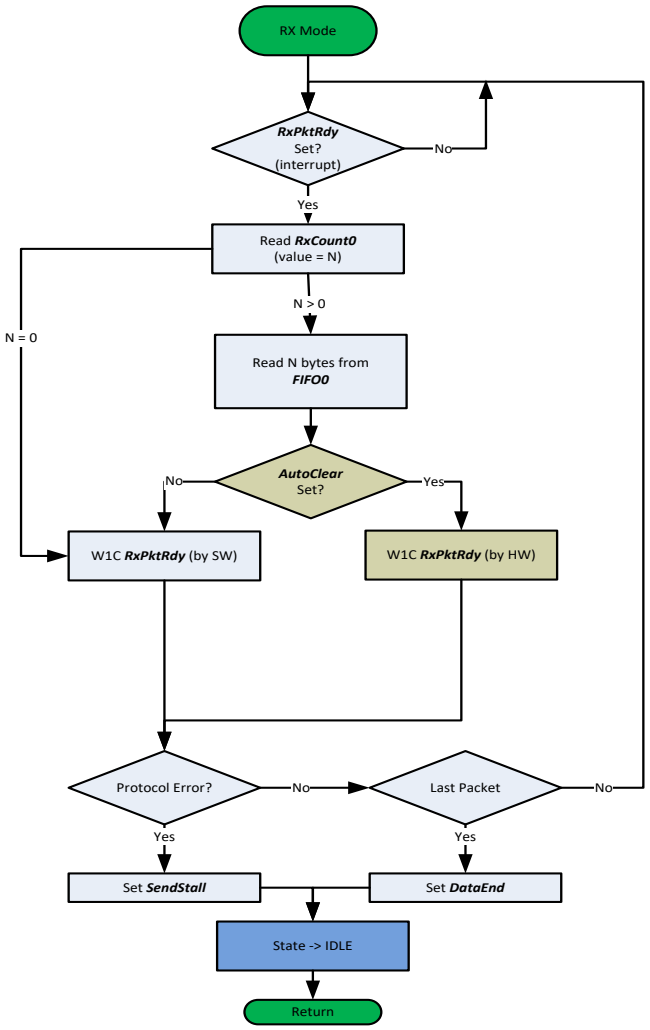


Figure 3-156 EP0 RX Mode Flow Chart

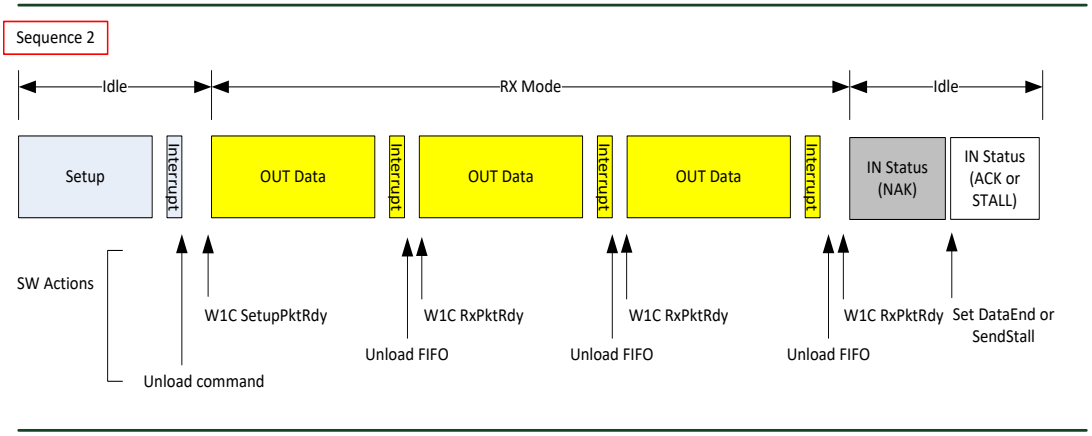


Figure 3-157 EP0 IDLE for RX Mode

3.12.5.7.4 QMU Top Programming Outline

The Queue Management Unit (QMU) is designed to unload software effort to serve DMA interrupts. By preparing GPD (USB) and the Buffer Descriptor (BD), software links data buffers and triggers the QMU to send/receive data to the host/from the device at a time.

3.12.5.7.4.1 GPD (USB) and BD Introduction

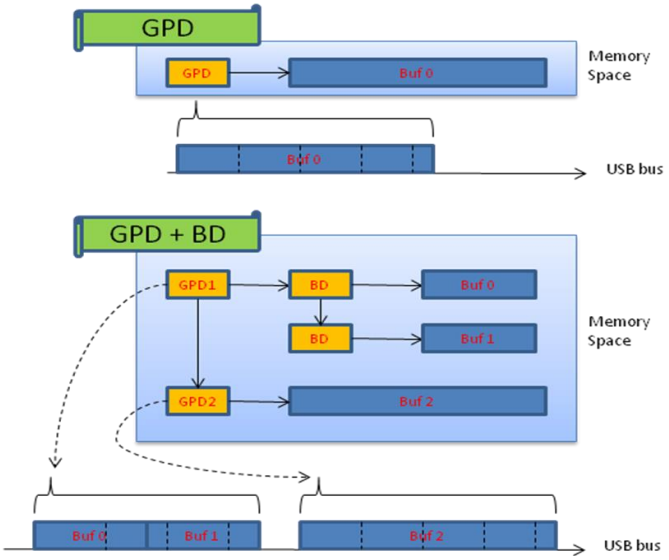


Figure 3-158 GPD (USB)/BD Bus Transfer

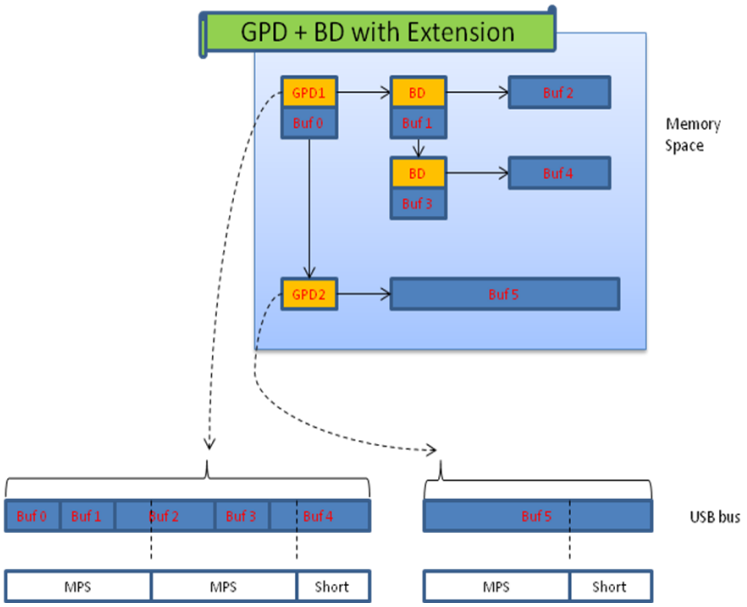


Figure 3-159 GPD (USB)/BD with Extension

Item	Description
BD	• Must contain a pointer pointing to a data buffer. • Capable of linking to another BD. • Capable of extension. Data is placed in memory immediately after this BD.
GPD (USB)	• Must contain a pointer pointing to a data buffer if this GPD (USB) does not link to any BD. • Capable of linking to several BDs. • Capable of extension. Data is placed in memory immediately after this GPD (USB). • Maps to a transfer on the USB. Data buffers are concatenated and transferred on the USB, data packet by data packet. Each data packet is smaller than Maximum Packet Size (MPS). • The order data buffers are concatenated.

Item	Description
	– GPD (USB) extension – GPD (USB) data buffer • For each BD in the chain – BD extension – BD data buffer

This section provides a brief description of BD and GPD (USB), with a comprehensive descriptor format presented in the following section for a more thorough understanding.

3.12.5.7.4.2 TX GPD (USB)/BD Format

Figure 3-160 shows the format of TX GPD (USB)/BD. For detailed description, refer to Sections 3.12.5.7.4.4 and 3.12.5.7.4.5.

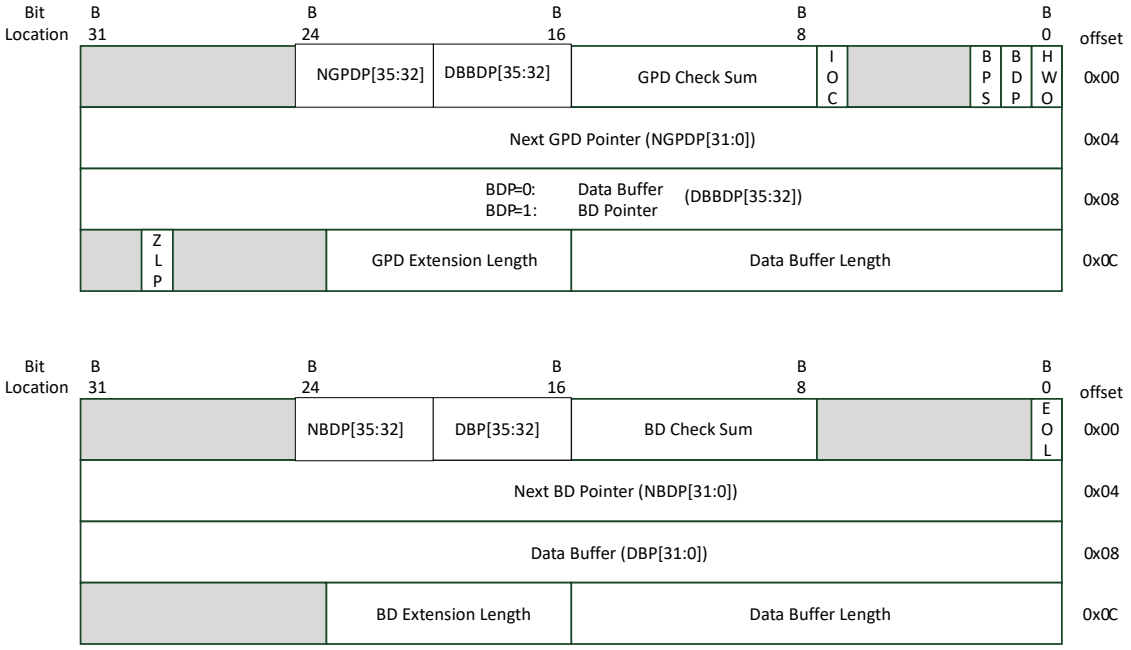


Figure 3-160 TX GPD (USB)/BD Format

3.12.5.7.4.3 RX GPD (USB)/BD Format

Figure 3-161 shows the format of RX GPD (USB)/BD. For detailed description, refer to Sections 3.12.5.7.4.4 and 3.12.5.7.4.5.

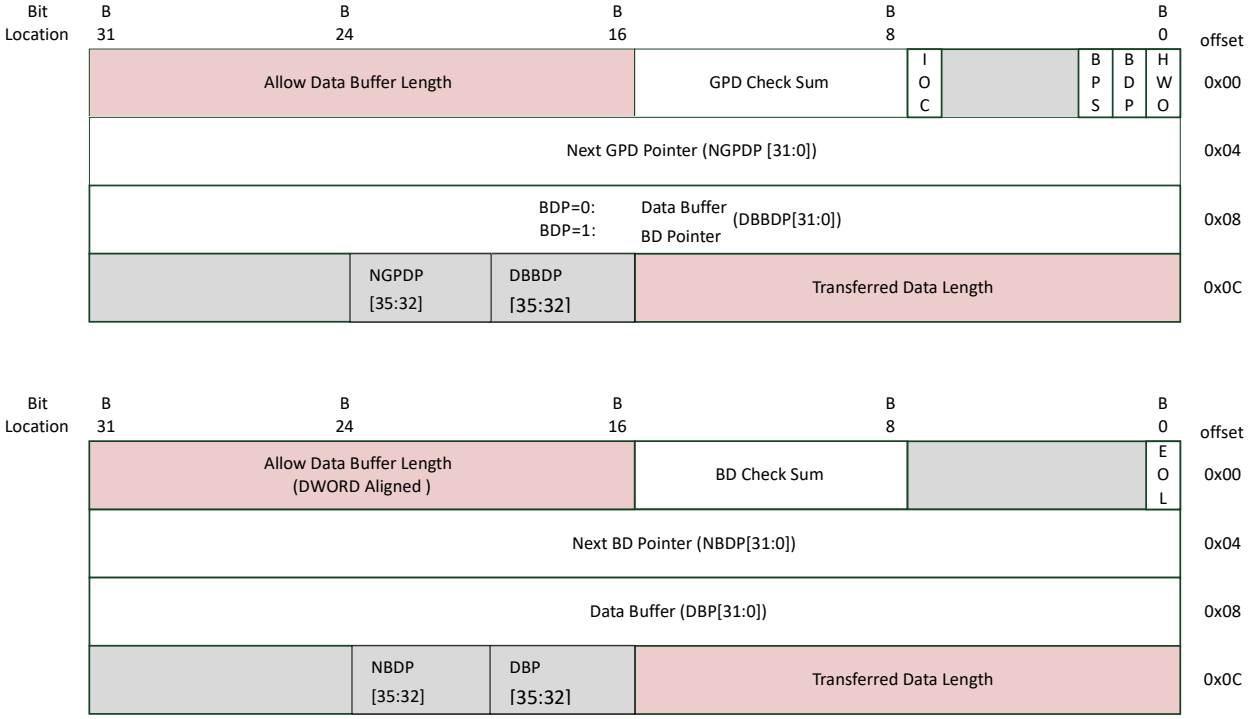


Figure 3-161 RX GPD (USB)/BD Format

3.12.5.7.4.4 GPD (USB) Field Description

Table 3-138 GPD (USB) Field Description

Field	Description
Hardware Ownership (HWO)	Indicates the current ownership of this GPD, the associated BD(s) and the associated data buffer(s). • 0: Software has the ownership. • 1: Hardware has the ownership.
Buffer Descriptor Present (BDP)	• 0: DWORD@0x8 points to a data buffer. • 1: DWORD@0x8 points to a BD.
Bypass (BPS)	• 0: Hardware does not skip this GPD if HWO = 1. • 1: Hardware skips this GPD if HWO = 1.
Interrupt On Completion (IOC)	• 0: Hardware does not issue an interrupt when this GPD (and the associated BDs) is completed. • 1: Hardware issues an interrupt when this GPD (and the associated BDs) is completed.
GPD (USB) Checksum	Validates the contents of this GPD (USB). • If TXQ_CS_EN/RXQ_CS_EN bit is set, an interrupt is issued when checksum validation fails. • Q_CS16B_EN decides the way in which the checksum value is calculated. – 0: Over the first 12 bytes of this GPD (USB) – 1: Over the first 16 bytes of this GPD (USB)
(RX only) Allow Data Buffer Length	Indicates the length of the assigned data buffer.
Next GPD(USB) Pointer	• Value! = 0, pointing to the next GPD (USB).

Field	Description
	• Value = 0, not pointing to any GPD (USB).
Data Buffer/BD Pointer	Refer to BDP description.
(TX only) Data Buffer Length	Indicates the length of the assigned data buffer.
(RX only) Transferred Data Length	• After receiving a transfer, the total length of data is written to this field. • If the total length of data is over 64 K, '0' is written to this field. Software has to sum up all BD "Transferred Data Length" to get the total length of data.
(TX only) GPD (USB) Extension Length	• 0: Not to use the GPD (USB) extension feature. • 1-255: Specifies the GPD (USB) extension buffer size. GPD (USB) extension buffer is placed in memory immediately after this GPD (USB). • Zero Length Packet (ZLP)

3.12.5.7.4.5 BD Field Description

Table 3-139 BD Field Description

Field	Description
End of List (EOL)	• 0: Not the last BD in the chain. The next BD is pointed by "Next BD Pointer". • 1: The last BD in the chain.

B Checksum	Validates the contents of this BD.
	• If TXQ_CS_EN / RXQ_CS_EN bit is set, an interrupt is issued when checksum validation fails.
	• Q_CS16B_EN decides the way that checksum value is calculated.
	  – 0: Over the first 12 bytes of this BD.
	  – 1: Over the first 16 bytes of this BD.

(RX only) Allow Data Buffer Length	Indicates the length of the assigned data buffer.
Next BD Pointer	Point to the next BD. Refer to EOL description.
Data Buffer	Point to data buffer
(TX only) Data Buffer Length	Indicates the length of the assigned data buffer.
(RX only) Transferred Data Length	After receiving a transfer, the length of data transferred to the data buffer is written to this field.
(TX only) BD Extension Length	• 0: Not to use BD extension feature. • 1-255: Specifies the BD extension buffer size. BD extension buffer is placed in memory immediately after this BD.

3.12.5.7.5 TXQ Programming Flow

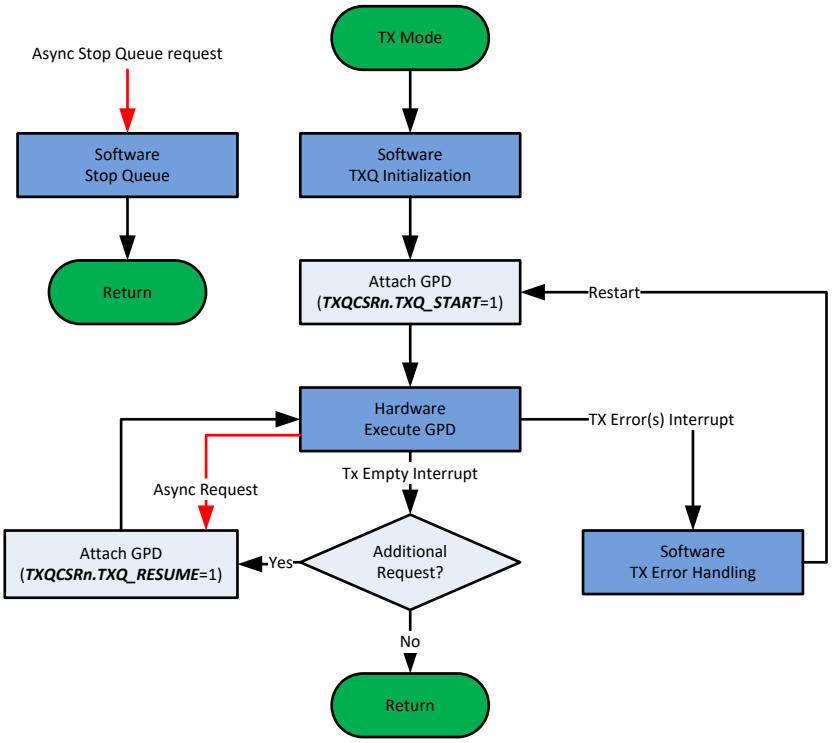


Figure 3-162 TXQ Programming Flow

3.12.5.7.5.1 TXQ Initialization Flow

Refer to *"MT8395 Register Map"* for detailed register descriptions.

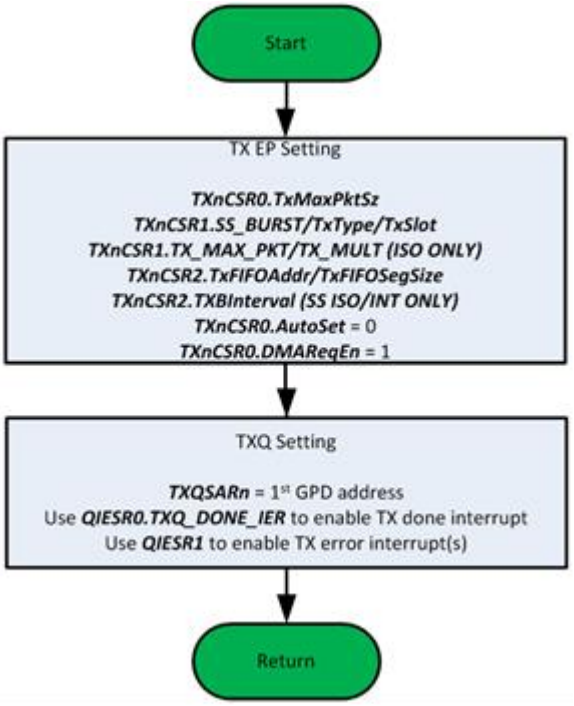


Figure 3-163 TXQ Initialization Flow

3.12.5.7.5.2 TXQ Stop Queue Flow

During the operation of TXQ, certain scenarios may arise that require the software to halt the current transfer. In such cases, there are two notification methods employed to inform the host.

- The first method involves explicitly notifying the host by STALLing EP.
- The second method involves implicitly notifying the host by sending a short packet to terminate the current transfer.

Regardless of the chosen notification method, it is essential for the host to perform error handling, and the device side must initiate the TXQ restart procedure based on the specific application requirements.

3.12.5.7.5.3 TXQ STALL (Recommended)

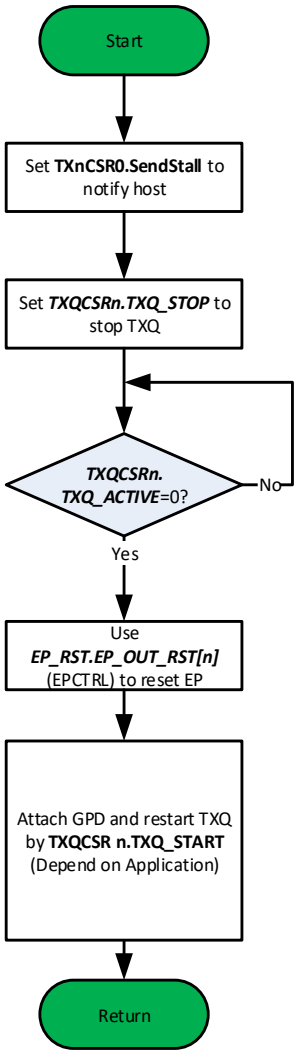


Figure 3-164 TXQ STALL Flow

3.12.5.7.5.4 TXQ Short Packet (I)

In this scenario, the host receives a ZLP, leading to the termination of the IN transfer. As EPOUT_RST is not utilized, data is fetched before the TXQ is halted. Upon the resumption of the TXQ, the data may be transmitted to the host as required.

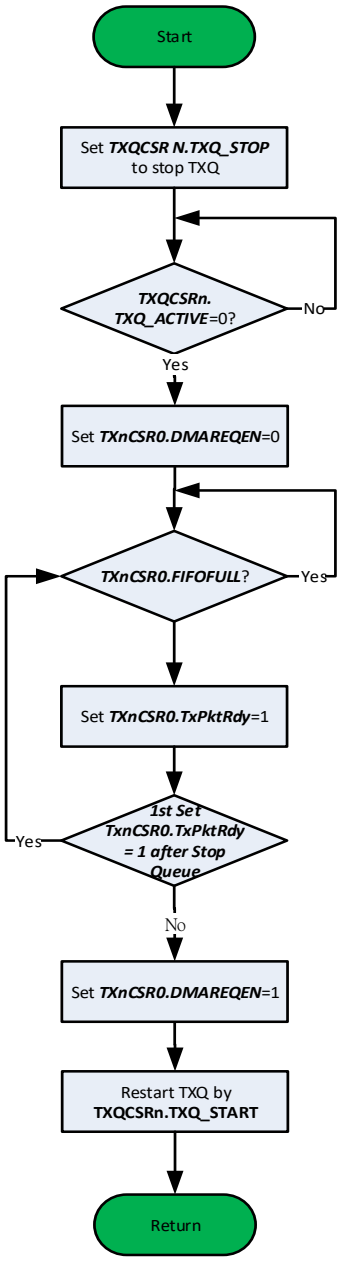


Figure 3-165 TXQ Short Packet (I) Flow

3.12.5.7.5.5 TXQ Short Packet (II)

In this scenario, the device software must ensure that the short packet aligns with the description specified in GPD (USB) to allow the host to receive the short packet and terminate the IN transfer. As EPOUT_RST is not employed, the data is fetched before the TXQ is halted. Upon the resumption of the TXQ, the data may be transmitted to the host as required.

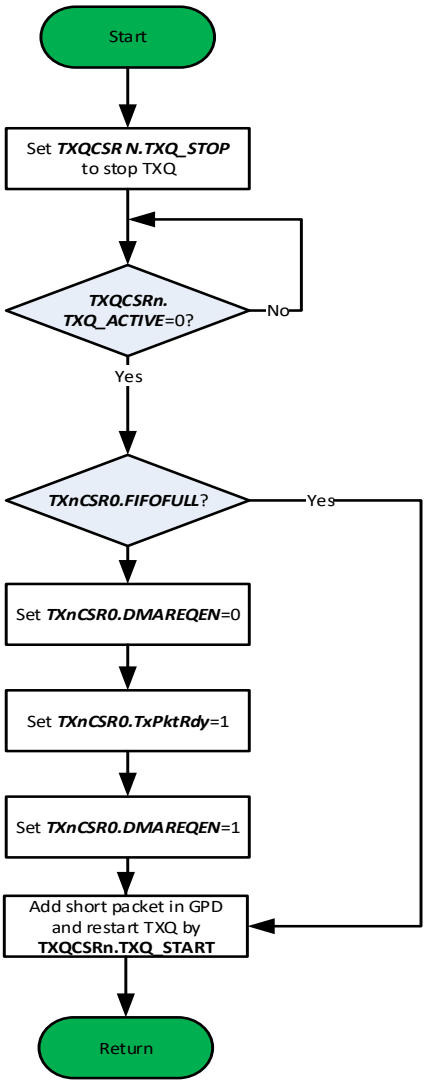


Figure 3-166 TXQ Short Packet (II) Flow

3.12.5.7.5.6 TXQ Error Handling Flow

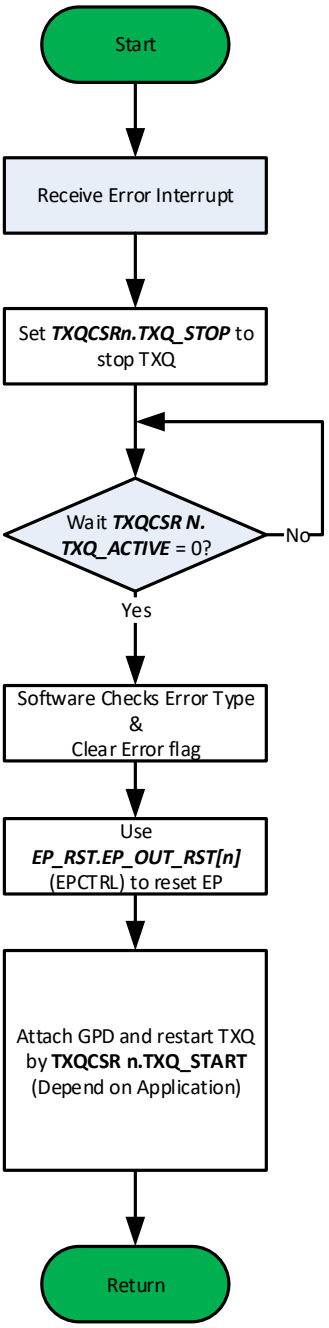


Figure 3-167 TXQ Error Handling Flow

3.12.5.7.5.7 TXQ GPD (USB) Execution Flow

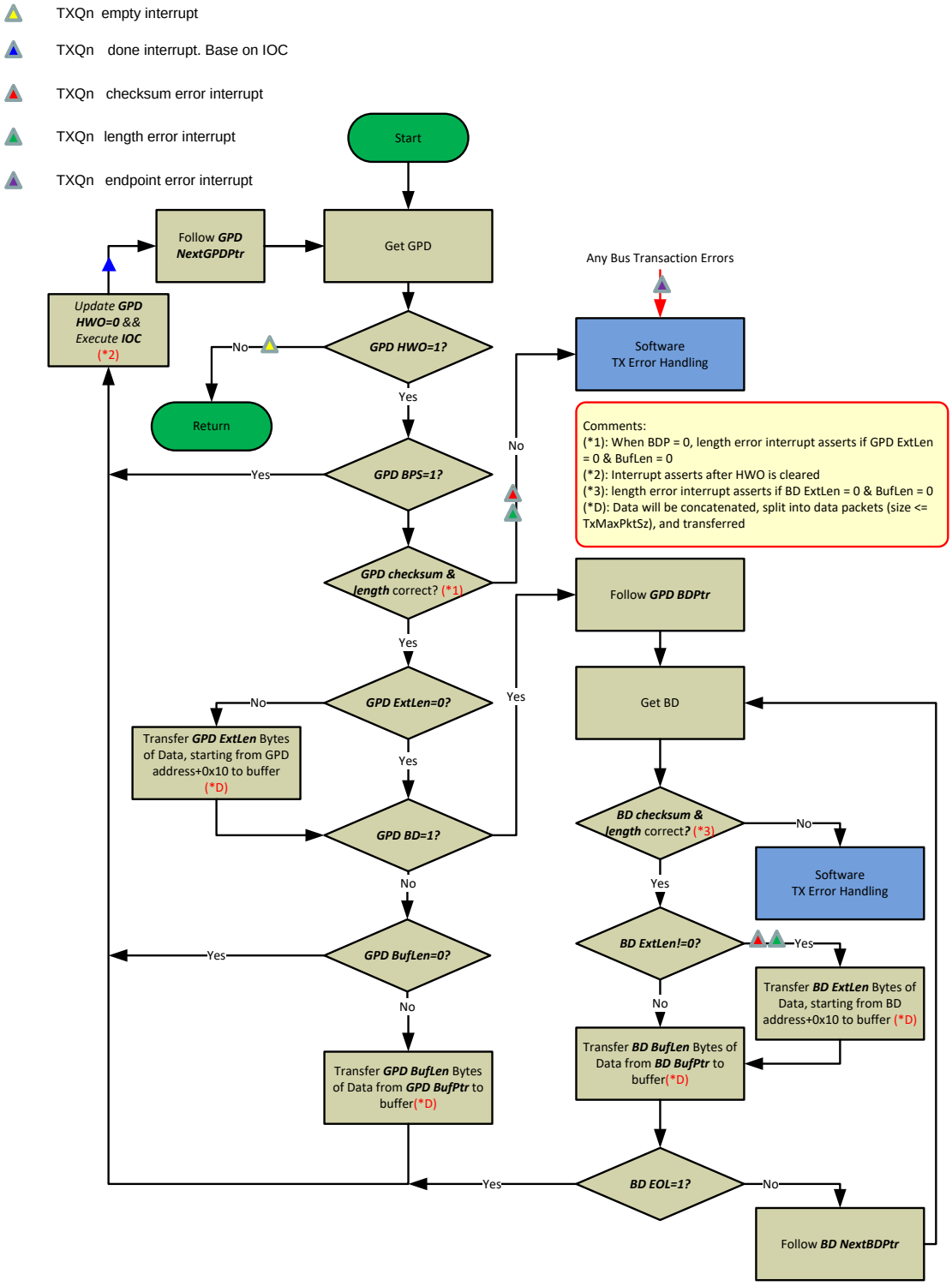


Figure 3-168 TXQ GPD (USB) Execution Flow

Note that if a "bus transaction error" occurs, the TXQn endpoint error interrupt is asserted. The bus transaction error shall not occur in the normal case.

3.12.5.7.6 RXQ Programming Flow

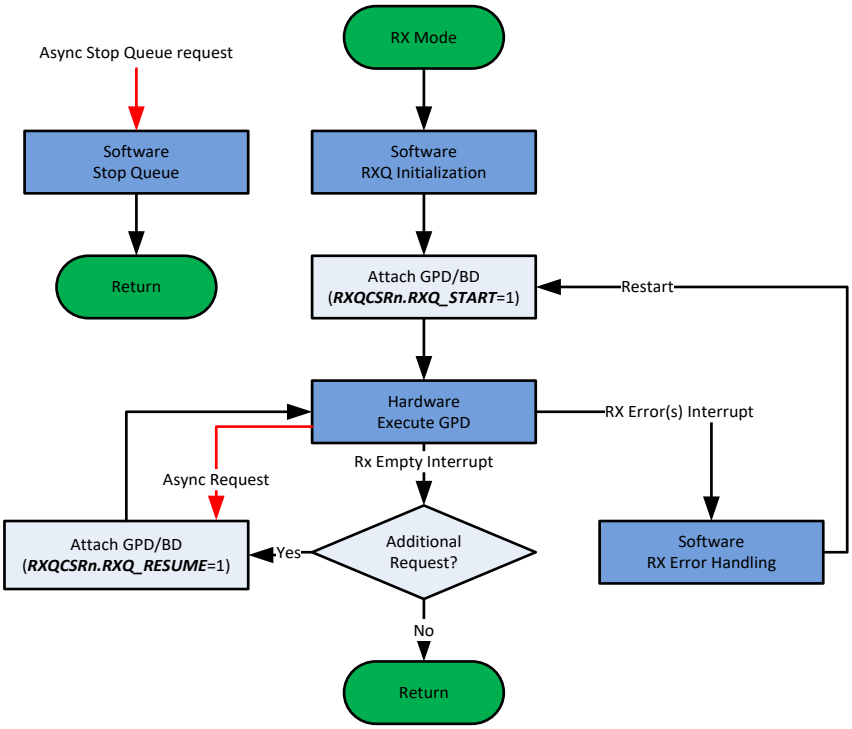


Figure 3-169 RXQ Programming Flow

Refer to *"MT8395 Register Map"* for detailed register descriptions.

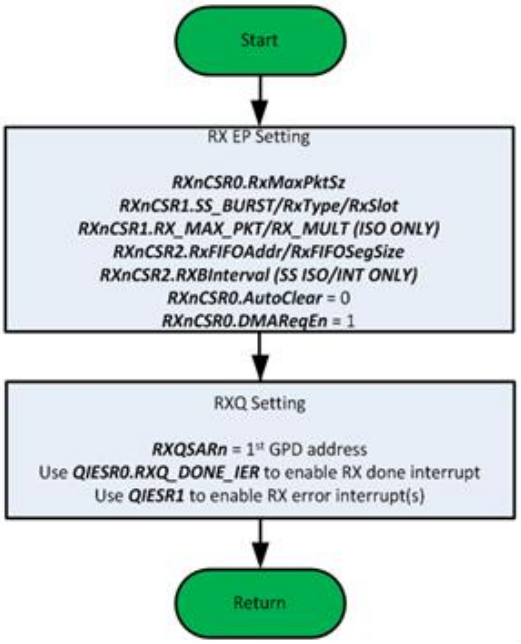


Figure 3-170 RXQ Initialization Flow

3.12.5.7.6.1 RXQ Stop Queue Flow

During the operation of RXQ, certain scenarios may arise that require the software to halt the current transfer. In such cases, there are two notification methods employed to inform the host.

- The first method involves explicitly notifying the host by STALLing EP.
- The second method involves silently dropping packets.

In the first case, the host must perform error handling. Upon the resumption of the RXQ, the device side initiates the restart procedures based on the specific application requirements.

3.12.5.7.6.2 RXQ STALL (Recommended)

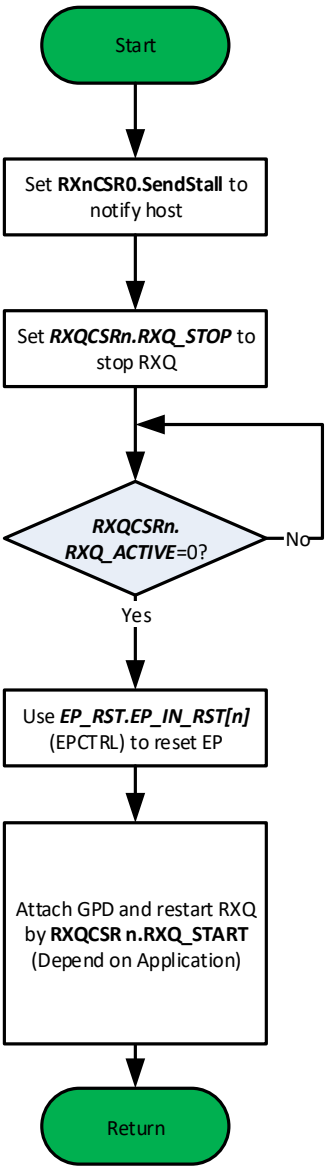


Figure 3-171 RXQ STALL Flow

3.12.5.7.6.3 RXQ Drop Packet

The device may experience reception errors due to the interruption and resumption of RXQ while the host is engaged in OUT transfer. During this process, some packets may be dropped, thereby corrupting the received data. To mitigate this issue, the device must perform error handling procedures.

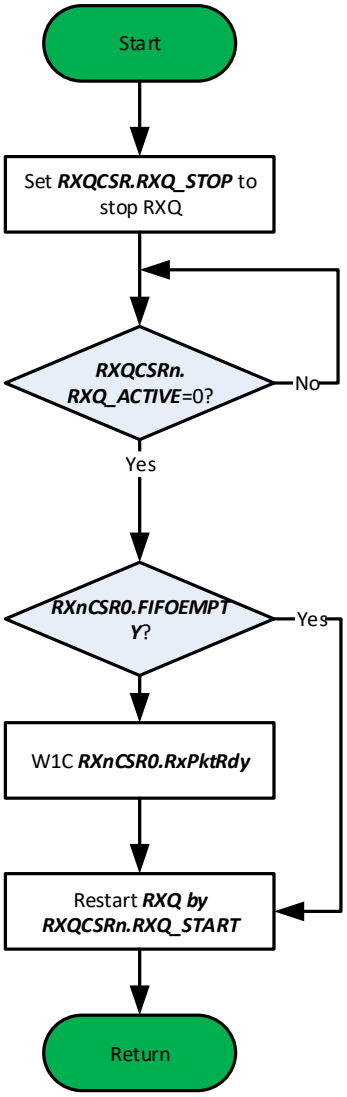


Figure 3-172 RXQ Drop Packet Flow

3.12.5.7.6.4 RXQ Error Handling Flow

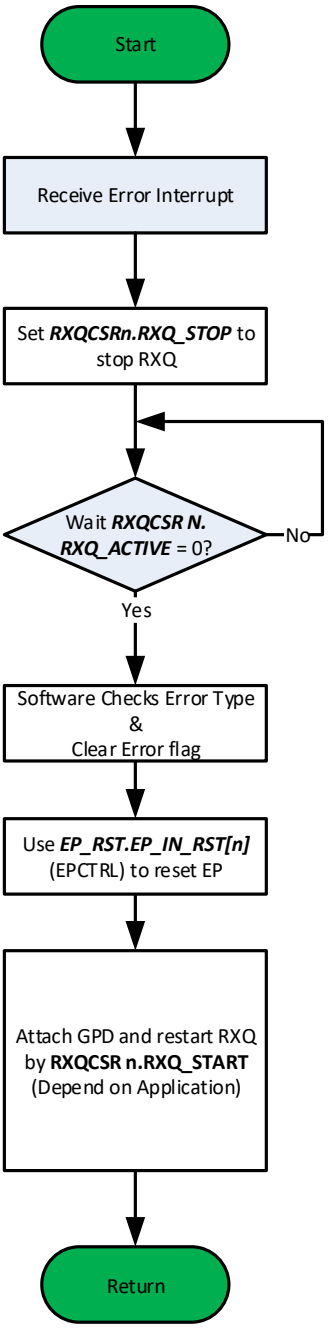


Figure 3-173 RXQ Error Handling Flow

3.12.5.7.6.5 RXQ GPD (USB) Execution Flow

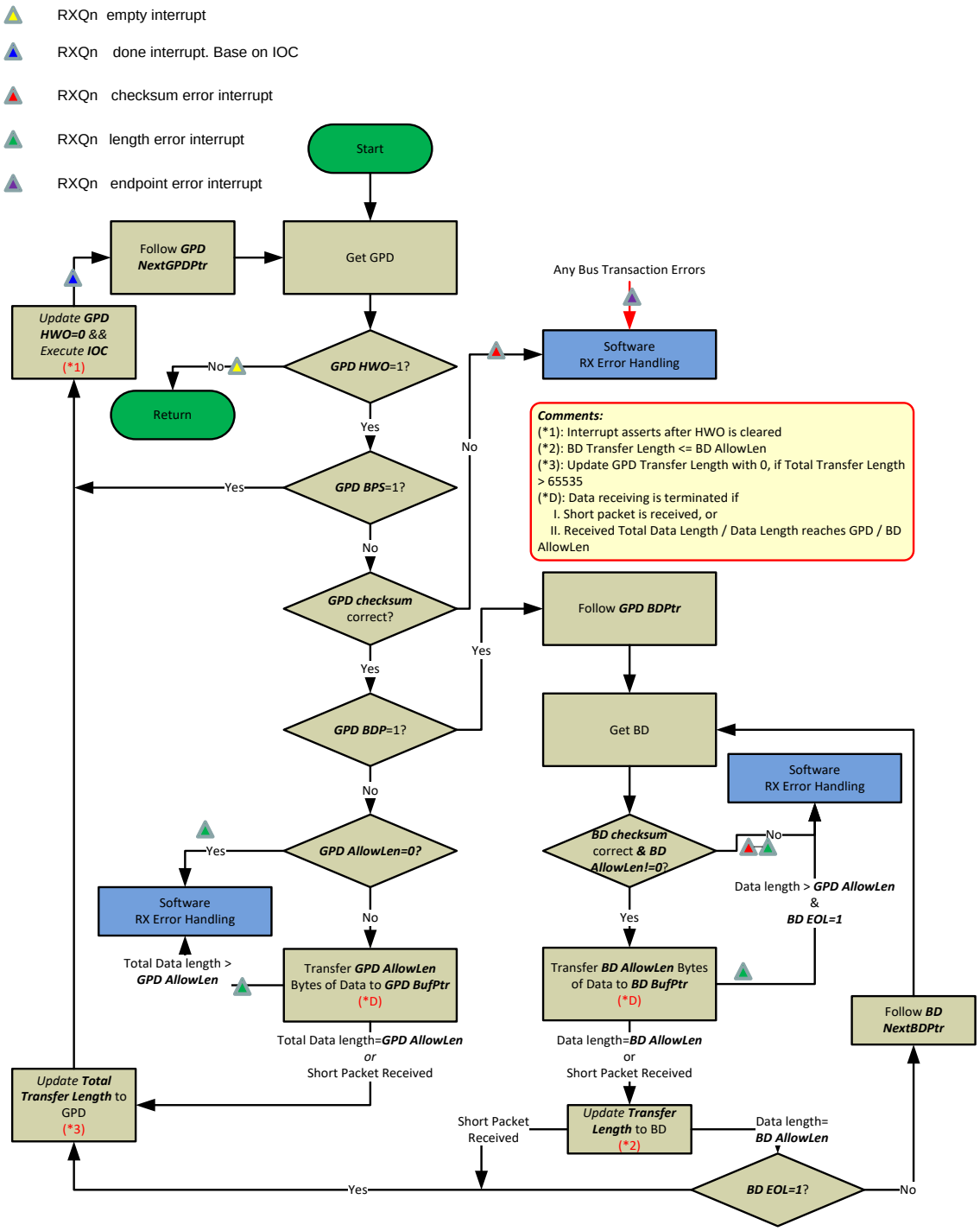


Figure 3-174 RXQ GPD (USB) Execution Flow

Note that if a "bus transaction error" occurs, the RXQn endpoint error interrupt is asserted. The bus transaction error shall not occur in the normal case.

3.12.5.8 Interrupt Top Programming Outline

1. **Adopt the multi-level interrupt architecture.**

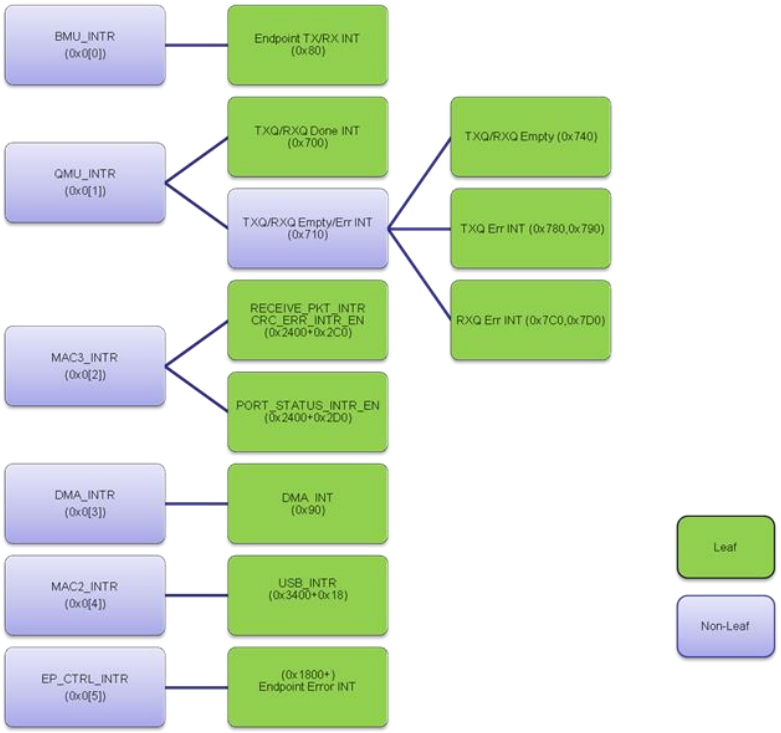


Figure 3-175 Interrupt Architecture

Refer to *"MT8395 Register Map"* for interrupt occurrence conditions.

Generally, there are two kinds of interrupt occurrence conditions:

Condition	Description
Endpoint TX/RX INT (0x80)	For the USB device, each EP has its own independent interrupt status bit, in order to indicate which EP interrupt occurs.
TXQ Err INT (0x780,0x790), RXQ Err INT(0x7c0,0x7d0)	The USB device does the transfer by QMU block. At QMU, software needs to fill GPD (USB) or BD to queue transfer. TXQ Err INT (0x780,0x790): • TX Queue GPD (USB)/BD Checksum Error • TX Queue GPD (USB) Data Buffer Length Error • RXQ Err INT (0x7c0,0x7d0): • RX Queue Generic Packet Descriptor Checksum Error • RX Queue Generic Packet Descriptor Data Buffer Length Error

2. ISR (USB)

- Only ISR (USB) in "leaf" node can be W1C; ISR (USB) in "non-leaf" node is RU.
- "Leaf" node ISR (USB) should be W1C to clear ISR (USB) in "non-leaf" node after the corresponding events are served.

3. IER (Interrupt Enable Register)

The IER can be set/cleared by writing 1 to Interrupt Enable Set Register (IESER)/Interrupt Enable Clear Register (IECR), except for *MAC2_INTR/MAC3_INTR/EP_CTRL_INTR*.

- If the low level IER is not set, when an event happens, it is reflected in the low level ISR (USB), but not the high level ISR (USB) (as shown in the left side of the figure below).
- If the low level IER is set, when an event happens, it is reflected in the low level ISR (USB) and high level ISR (USB) (As shown in the right side of the figure below)

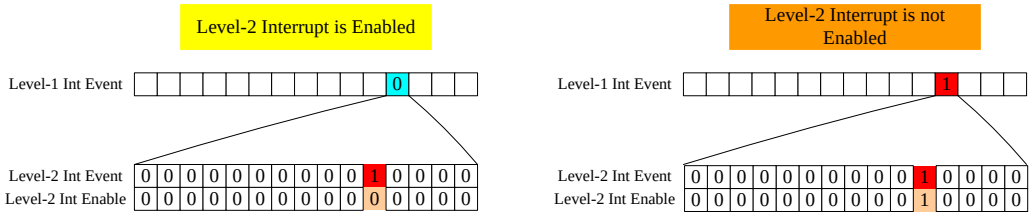


Figure 3-176 Example Interrupt Setting

3.12.5.8.1 Queue Management Unit (QMU) Interrupt Architecture

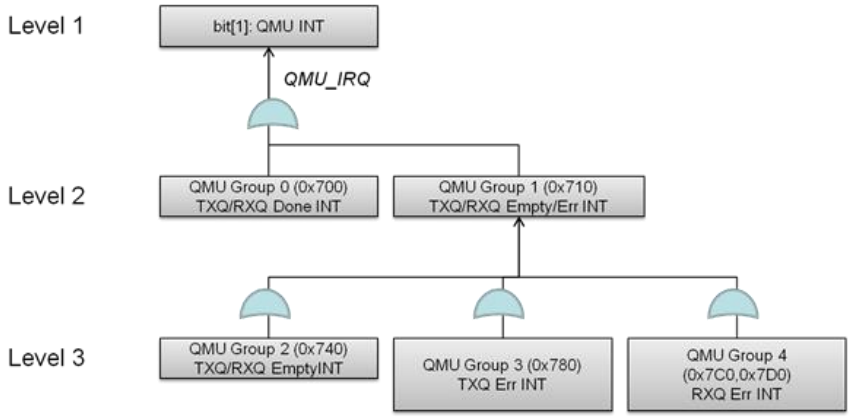


Figure 3-177 QMU Interrupt Layer

A queue has different behaviors for different TX/RX interrupts.

Interrupt	Behavior
TX	• A queue does not stop when – A queue is completed when GPD is done with IOC = 1. • A queue stops when – A queue is empty. – There is a queue, checksum or length error.
RX	• A queue does not stop when – A queue is completed when GPD is done with IOC = 1. – There is a queue or ZLP error. • A queue stops when – A queue is empty. – There is a queue, checksum or length error.

The figure below depicts the related status/mask/mask set/mask clear registers for each group.

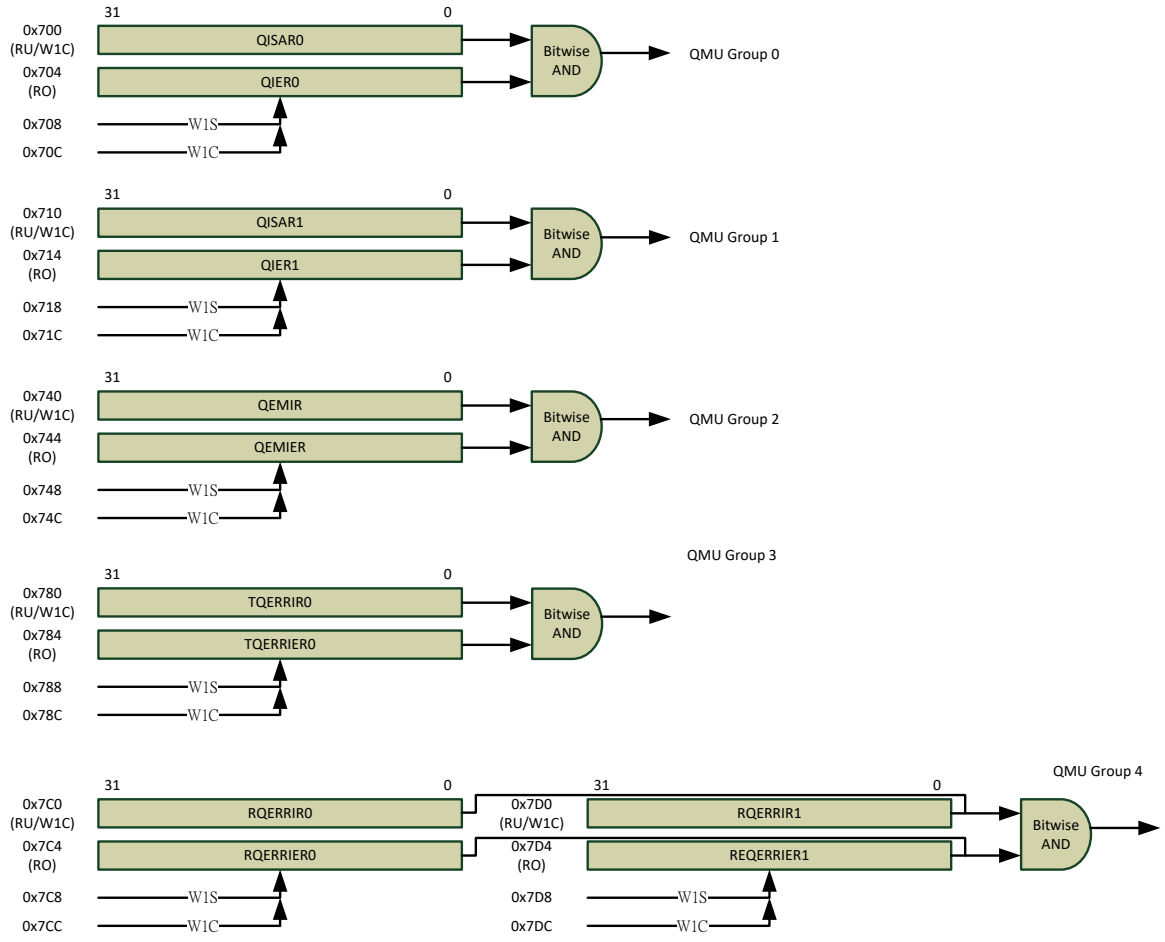


Figure 3-178 Interrupts Status/Mask/Mask Set/Mask Clear

3.12.5.9 Power Saving Scenario

In order to conserve power, USB provides registers for software to power down the unnecessary hardware parts.

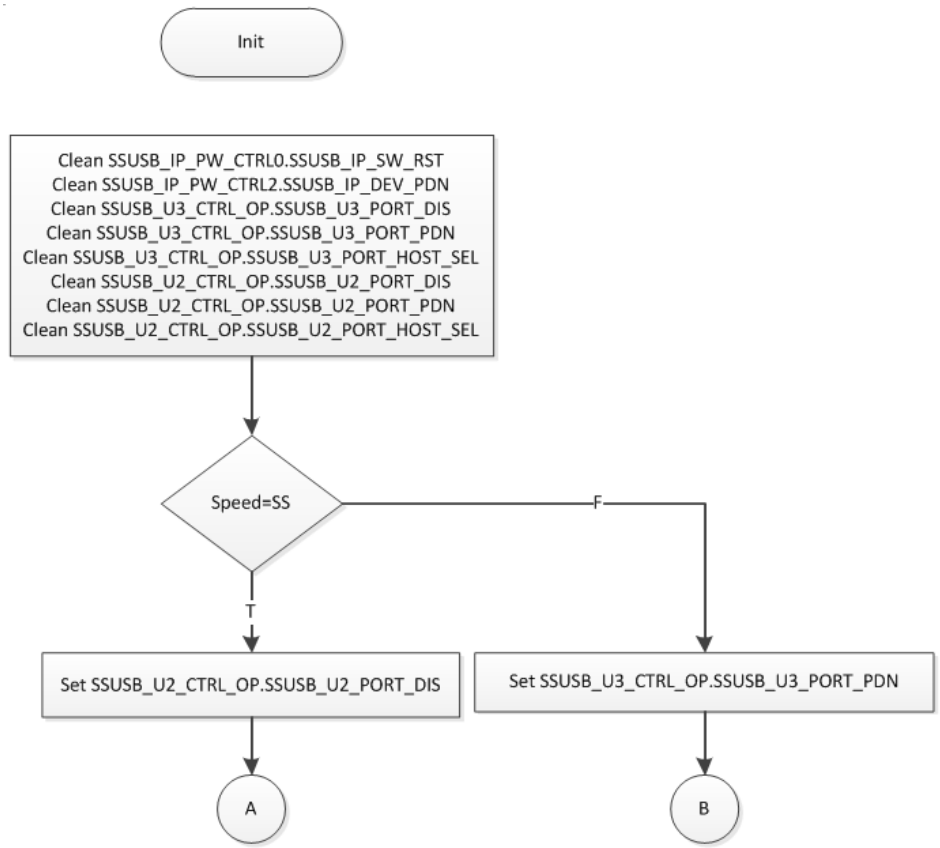


Figure 3-179 Power Saving Flow: Initial

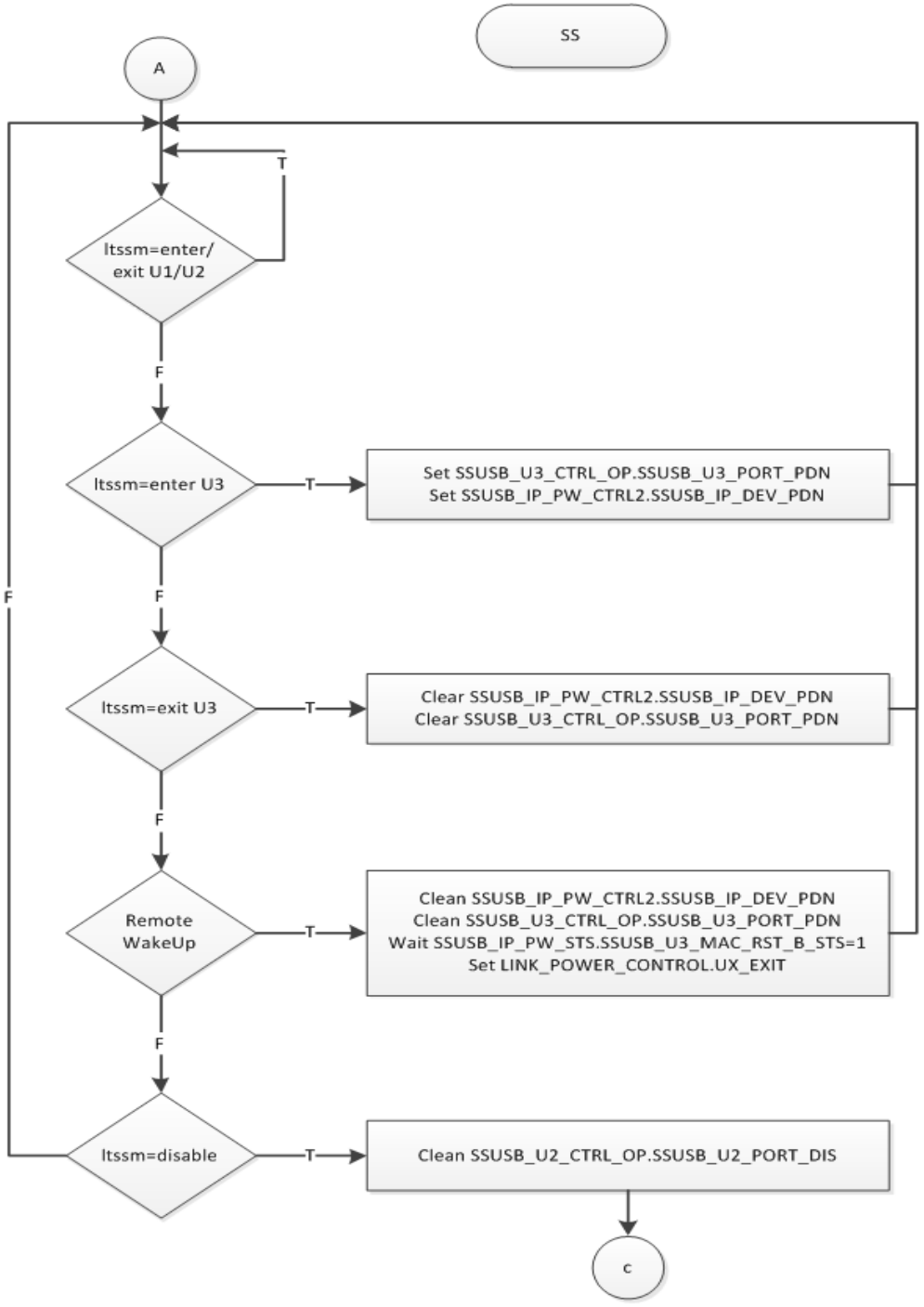


Figure 3-180 Power Saving Flow: SuperSpeed

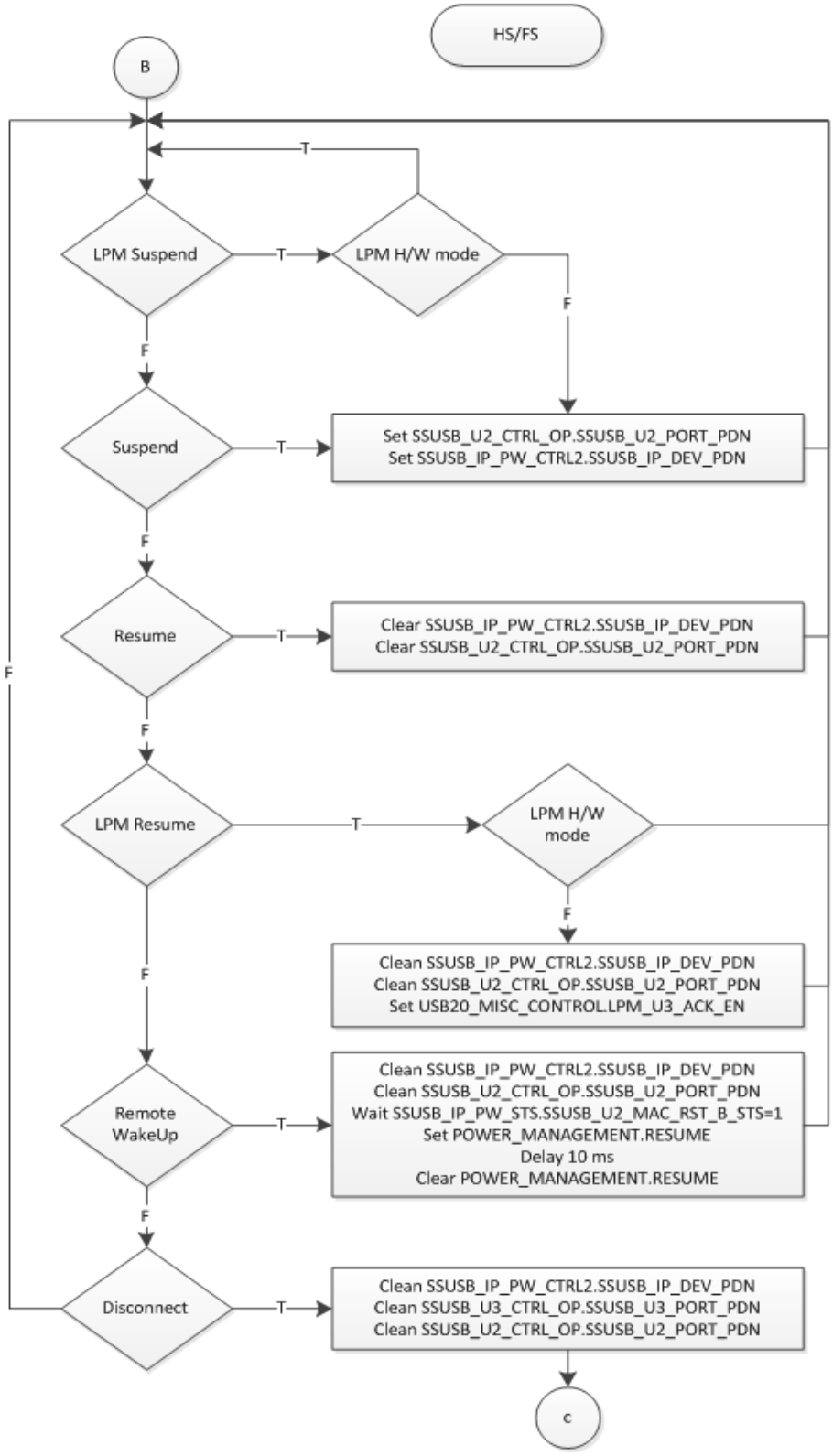


Figure 3-181 Power Saving Flow: High-speed and Full-speed

3.12.5.9.1 LPM Recommended Setting

- *lpm_mode = 0* and *lpm_hrwe = 0*: Normal LPM mode, enable hardware remote wakeup.
- SSUSB_DEV.MAC_U2_EN_CTRL [20:16] = 0x1f: Enable LPM to accept check and accept LPM request when all QMU/EP are inactive.

For Bulk transfer:

- OUT EPs: Enable L1_EXIT_EP_OUT_CHK and L1_EXIT_EP_OUT_FC_CHK.
- IN EPs: Enable L1_EXIT_EP_IN_CHK and L1_EXIT_EP_IN_FC_CHK.
- EP0: Enable L1_EXIT_EP0_CHK and L1_EXIT_EP0_FC_CHK.

Therefore, the device can resume the bus as long as EP FIFO is ready, and the flow control assures that the EP has data to transfer or accept.

For periodic (isochronous/interrupt) EPs, the host can easily predict the point to send/receive data. Therefore, it is better for the host to determine the time point to resume the bus and process transfers. It is suggested to disable *L1_EXIT_EP_OUT_CHK* and *L1_EXIT_EP_IN_CHK* for periodic EPs.

3.12.5.10 USB Device Reset Mode

Table 3-140 Reset Sequence of USB Device Mode

Step	Address	Register Name	Local Address	R/W	Value	Description
Toggle USB software reset						
1	ssusb_sifslv_ippc_Base address + 0x0000	SSUSB_IP_PW_CTRL0	SSUSB_IP_SW_RST [0]	W	1'b1	USB Software Reset. When this bit is set, the whole USB is reset. Write “0” to release reset.
2	Delay 1µs					
3	ssusb_sifslv_ippc_Base address + 0x0000	SSUSB_IP_PW_CTRL0	SSUSB_IP_SW_RST [0]	W	1'b0	USB Software Reset. When this bit is set, the whole USB is reset. Write “0” to release reset.
Check reset status						
1	ssusb_sifslv_ippc_Base address + 0x0014	SSUSB_IP_PW_STS2	SSUSB_U2_MAC_SYS_RST_B_STS [0]	R		When this bit is 1'b1, it means that reset for mac2_sys_ck domain is inactive.
2	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_U3_MAC_RST_B_STS [16]	R		When this bit is 1'b1, it means that reset for mac3_mac_ck domain is inactive.
3	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_DEV_QMU_RST_B_STS [1]	R		When this bit is 1'b1, it means that reset for device QMU sys_ck domain is inactive.
4	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_DEV_BMU_RST_B_STS [2]	R		When this bit is 1'b1, it means that reset for device BMU sys_ck domain is inactive.
5	ssusb_sifslv_ippc_Base address + 0x0010	SSUSB_IP_PW_STS1	SSUSB_DEV_RST_B_STS [3]	R		When this bit is 1'b1, it means that reset for device excluding BMU and QMU sys125_ck. domain is inactive.

3.12.5.10.1 Address Reset

When the following conditions happen, the device address is reset to 0.

- USB 2.0

  - Upon receiving bus reset
- USB 3.1 Gen1
  - When entering the “Polling” state
  - When receiving a warm Reset, and then entering the “Polling” state.
  - When receiving a hot reset.

3.12.5.10.2 EP Reset

The EPs are reset by hardware when the following conditions happen.

- USB 2.0
  - Upon receiving bus reset
  - Reset
    - Data toggle
    - FIFO pointer (FIFO address register is not touched)
    - EP flow control status
- USB 3.1 Gen1
  - When entering the “Polling” state
  - When receiving a warm reset, and then entering the “Polling” state.
  - When receiving a hot reset.
  - Reset
    - EP sequence number
    - EP flow control status
    - EP packet pending status
    - EP packet counter
    - EP active status
    - FIFO pointer (FIFO address register is not touched.)

Since the following commands may affect the EP application and its configuration.

- SET_CONFIGURATION
- SET_INTERFACE
- CLEAR_FEATURE ENDPOINT_HALT

Software can do the following operations before EP re-configuration:

- Use bits in the register of EP_RST to reset each EP.
- Stop the corresponding queue(s).

3.12.5.11 EPn MCU Mode Top Programming Outline (BULK Interrupt Mode Only)

3.12.5.11.1 EPn MCU Mode TX Programming Flow

- *TXnCSR0.TxMaxPktSz* defines the maximum packet size (in bytes). Software can load data into FIFOn at a time.
- IN transfer flow
  a. Software checks whether FIFOn is full. If not, software loads a data packet to FIFOn, and sets *TXnCSR0.TxPktRdy*.
  b. After data packet is sent to the host, an interrupt is generated to notify software to repeat the operation until all data is sent to the host.

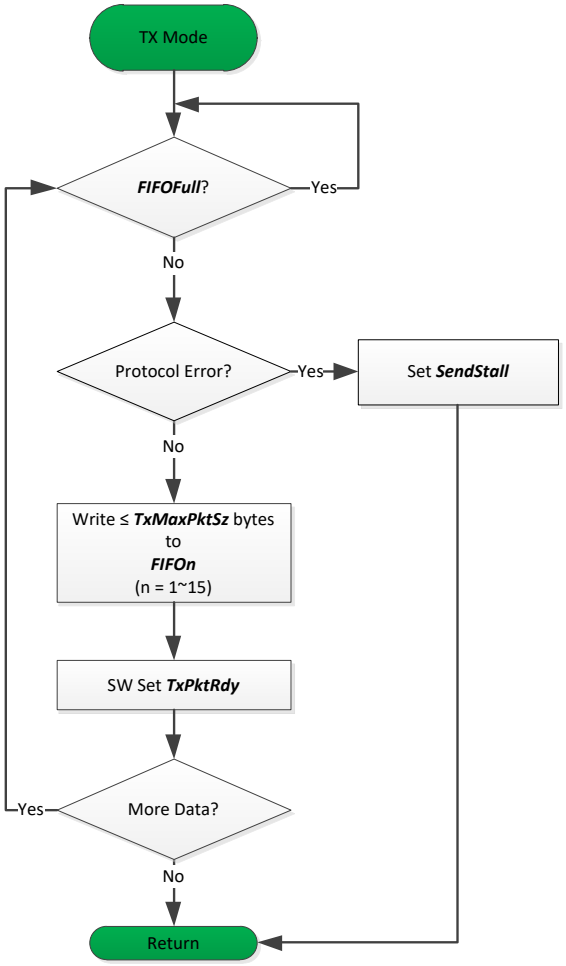


Figure 3-182 EPn MCU Mode TX Programming Flow

3.12.5.11.2 EPn MCU mode RX Programming Flow

- *RXnCSR0.RxMaxPktSz* defines the maximum packet size (in bytes). Software can load data from FIFOn at a time.
- OUT transfer flow:
  1. *RXnCSR0.RxPktRdy* is set and an interrupt is generated to notify software.
  2. Software reads RXnCSR3.EP_RX_COUNT, unloads RXnCSR3.EP_RX_COUNT bytes from FIFOn and W1C RXnCSR0.RxPktRdy.
  3. Software repeats operation until there is no more data.

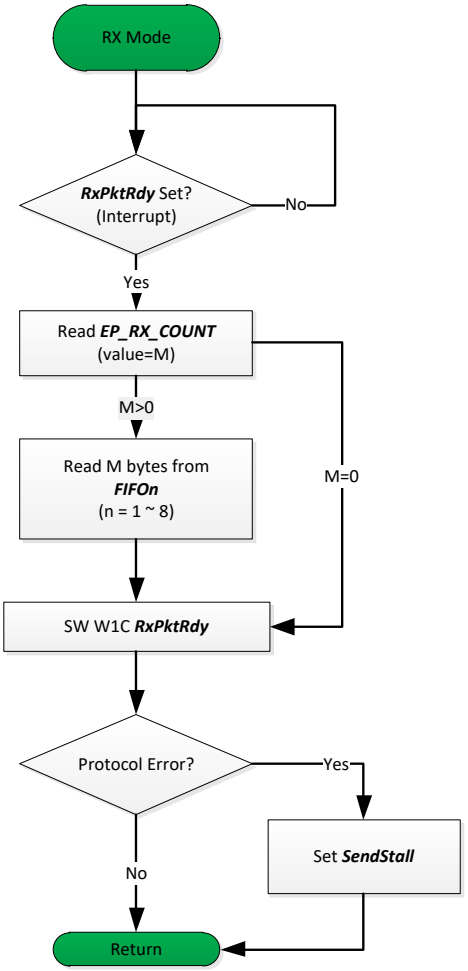


Figure 3-183 EPn MCU Mode RX Programming Flow

3.12.5.12 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.12.5.13 References

- Protocol
  - Universal Serial Bus 3.1 Specification, Revision 1.0, July 26, 2013
  - Universal Serial Bus Specification, Revision 2.0, April 27, 2000
  - Extensible Host Controller Interface (xHCI) for Universal Serial Bus, Revision 1.1,12/20/2013
- PHY
  - PHY Interface for the PCI Express, SATA, and USB 3.1 Architectures, Version 4.3
  - UTMI+ Specification, Revision 1.0, February 25th, 2004
- Bus
  - AMBA AXI and ACE Protocol Specification AXI3, AXI4, and AXI4-Lite ACE and ACE-Lite, 28 October 2011

3.12.5.14 USB 3.1 PHY Overview

The PHY is responsible for managing the low-level protocol and signaling functions. This includes data serialization and de-serialization, 8b/10b encoding/decoding (5 Gbps), analog buffers, elastic buffers and receiver detection.

The controller interfaces with the PHY via a PIPE (SuperSpeedPlus) interface. The PIPE supports 32 bits * 125 MHz (5 Gbps).

3.12.5.15 Features

- Fully compliant with USB 3.1 Specification
- Electrical sub-blocks support 5.0 GT/s operation, which is compliant with USB 3.1 electrical specification.
- Fully compliant with PHY interface for the USB Architectures (PIPE), Version 4.3
- Utilizes the 32-bit parallel interface for 5.0 GT/s.

3.12.5.16 Block Diagram

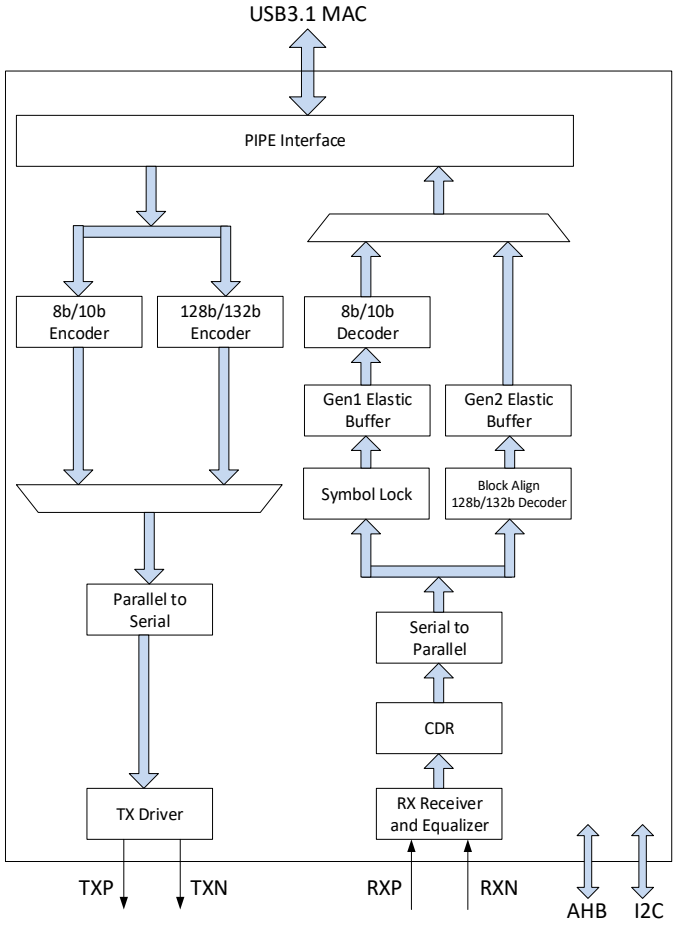


Figure 3-184 Block Diagram of USB 3.1 Gen1 PHY

Figure 3-184 displays the USB3.1 PHY block diagram, which comprises two main sub-modules: the Physical Layer Analog Block (PHYA) and the Physical Layer Digital Block (PHYD).

- The PHYA includes:

- The TX driver, which outputs differential pair signals.
- The RX front-end, which receives differential pair data.
- The CDR, which recovers timing information from the serial data stream.
- Serializer and de-serializer, which convert data between the serial and parallel data interfaces in each direction.

- The PHYD includes:
  - 8b/10b encoder and decoder (for 5.0 GT/s)
  - Elastic buffers, which compensate for differences in frequencies between bit rates at the ends of a link.

For the FT/debug mode, the I2C interfaces (*sif_scl, sif_sdin, sif_sdout*, and *sif_sden*) of the *PEXTP_PHY_TOP* connection should be verified and controlled through the pin. In addition, in the FT/debug mode, the Schmitt trigger should be enabled at the pads of I2C SDA/SCL. Refer to Figure 3-185 for the I2C signal connection.

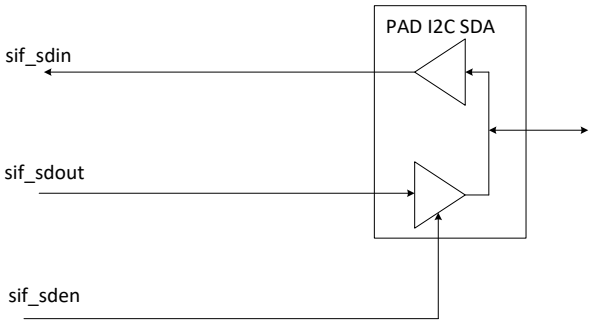


Figure 3-185 Connection of I2C Signals

3.12.5.17 Electrical Characteristics

The SSUSB electrical characteristics are compatible with USB 3.1 Specification Revision 1.1.

Table 3-141 USB 3.1 Spread Spectrum Clock (SSC) Electrical Characteristics

Description	Min.	Typ.	Max.	Unit
SSC modulation rate	30	-	33	kHz
SSC deviation	+0/-4000 (Gen 1)	-	+0/-5000(Gen 1)	ppm

Note:

- Refer to USB 3.1 Specification Revision 1.1, Section 6.5.4, Table 6-17.

Table 3-142 USB 3.1 Gen1 Transmitter and Receiver Electrical Characteristics

Description	Min.	Typ.	Max.	Unit
Transmitter Parameters				
Data rate	10			GT/s
Unit interval*	199.94 (Gen 1)	-	200.06 (Gen 1)	ps
TX differential peak to peak voltage swing	0.8	1	1.2	V
TX eye width	0.625 (Gen 1)	-	-	UI
DC differential TX impedance	72	-	120	Ω

Description	Min.	Typ.	Max.	Unit
TX AC common mode voltage active	-	-	100	mV
SSC maximum slew rate	-	-	10	ms/s
Receiver Parameters				
LFPS detect threshold	100	-	300	mV
DC differential RX impedance	72	-	120	Ω

Note:

- Do not account for SSC caused variations. Refer to USB 3.1 Specification Revision 1.1, Section 6.7.1, Table 6-18, Table 6-19.

3.12.5.18 Interface Signal

3.12.5.18.1 Analog Pads

Signal Name	Input/Output	Description
PAD_SSUSB_RXN_P1/2	Inout	Analog PAD, Lane0/1 RX-
PAD_SSUSB_RXP_P1/2	Inout	Analog PAD, Lane0/1 RX+
PAD_SSUSB_TXN_P1/2	Inout	Analog PAD, Lane0/1 TX-
PAD_SSUSB_TXP_P1/2	Inout	Analog PAD, Lane0/1 TX+
PAD_XTP_GLB_CKP	Inout	Analog PAD, Ref 100MHz CK+
PAD_XTP_GLB_CKN	Inout	Analog PAD, Ref 100MHz CK-

3.12.5.18.2 Clock and Testing

Signal Name	Input/Output	Description	Note
ANA_OLT	Input	Stress test mode	Reserved. Tied to 0.
ANA_SCAN	Input	Analog scan signal	
AD_XTP_GLB_*_OLT_*	Output	Monitor analog signal by LED	
DA_XTP_GLB_*_SCAN_*		Analog scan signals	
ADA_XTP_GLB_LDO_VREF	Input	Analog LDO reference voltage for HTOL.	Provide HTOL VREF if needed. Otherwise, connect to ADA_XTP_GLB_TIELO_12.
ADA_XTP_GLB_TIELO_12	Output	Analog low signal for unused input-type ADA_*.	
ADA_XTP_GLB_MOUNT	Inout	Analog debug signal	
ADA_XTP_GLB_POR_RSTB_15	Inout	Analog power-on reset	1.5V power domain
ADA_CKM_XTAL_CK	Input	Ref 26 MHz CK from analog module	Provide 26 MHz clock if needed. Otherwise, connect to ADA_XTP_GLB_TIELO_12.
DA_CKM_XTAL_CK	Input	Ref 26 MHz CK from digital module	Provide 26 MHz clock if needed. Otherwise, tie 0.
scan_mode	Input	Scan Related	• 1: Scan mode. • 0: Function mode

Signal Name	Input/Output	Description	Note
scan_500m_ck	Input	Scan 500 MHz clock	500 MHz clock for ATPG test.
scan_312p5m_ck	Input	Scan 312.5 MHz clock	312.5 MHz clock for ATPG test.
scan_625m_ck	Input	Scan 625 MHz clock	625 MHz clock for ATPG test.
scan_enable	Input	Scan related	Scan shift enabled
scan_enable_cg	Input	Scan related	Clock gating enable during scan shifting.
ext_pwr_rst_b	Input	Power on reset	System reset. It should be combined with software reset.
sspxtp_sys_ck_type	Input	Reference clock type (ref_ck) 3'd0: 20 MHZ 3'd1: 24 MHZ 3'd1: 25 MHZ 3'd2: 26 MHZ 3'd3: 27 MHZ	
ref_ck	Input	Reference clock, should follow sspxtp_sys_ck_type setting frequency.	
sspxtp_ckm_en	Output	CKM control signal	
sspxtp_ckm_intck_req	Output	CKM request signal	
sspxtp_ckm_padck_req	Output	CKM request signal	
sspxtp_sleep	Output	USB low power state signal	
sspxtp_probe[15:0]	Output	SSUSB probe out signal	• For debug mode: Confirm whether sspxtp_dig_top/sspxtp_probe[15:0] can be output correctly. • For FT mode: Confirm whether sspxtp_dig_top/sspxtp_probe[15:0] can be output correctly.

3.12.5.18.3 I2C

Signal Name	Input/Output	Description	Note
sspxtp_i2c_mode	Input	SSUSB register control mode selection • 1'b0: AHB mode • 1'b1: I2C mode	• Should be 1'b0 by default in normal mode. Should be controlled by the chip • Should be set to 1'b1 in FT mode
sspxtp0_saddr[6:0]	Input	I2C device address	Request I2C device address. Device address is 7'h51.
sspxtp0_sclk	Input	I2C SCLK	Refer to Figure 3-185.
sspxtp0_sdin	Input	I2C SDIN	
sspxtp0_sdout	Output	I2C SDOUT	
sspxtp0_sden	Output	I2C SDEN	

3.12.5.18.4 AHB

Signal Name	Input/Output	Description	Note
mcu_bus_ck	Input	AHB interface clock for SSUSB	Connected to the CPU bus
sspxtp_phy_slv_hsel	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hwrite	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_htrans[1:0]	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hsize[1:0]	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_haddr[31:0]	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hwdata[31:0]	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hready_in	Input	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hresp	Output	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hready	Output	AHB interface	Connected to the CPU bus
sspxtp_phy_slv_hrdata[31:0]	Output	AHB interface	Connected to the CPU bus

3.12.5.18.5 PHY to MAC PIPE Interface

Signal Name	Input/Output	Description	Note
sspxtp_pipe0_rst_b	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_disable	Input	PIPE signal	Connect to SUB MAC
sspxtp_pipe0_rate	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_powerdown[1:0]	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_detectrx	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_elecidle	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_term	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_deem[17:0]	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_enc_dec_bypass	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_block_align_ctrl	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_eq_training	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_polarity	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_oneszeros	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_data_valid	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_sync_header[3:0]	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_start_block	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_datak[3:0]	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_tx_data[31:0]	Input	PIPE signal	Connect to USB MAC

Signal Name	Input/Output	Description	Note
sspxtp_pipe0_tx_data[31:0]	Input	PIPE signal	Connect to USB MAC
sspxtp_pipe0_phystatus	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_valid	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_elecidle	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_status[2:0]	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_sync_header[3:0]	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_data_valid	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_start_block	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_data[31:0]	Output	PIPE signal	Connect to USB MAC
sspxtp_pipe0_rx_datak[3:0]	Output	PIPE signal	Connect to USB MAC

3.12.6 Ethernet Network Interface Controller (ENIC)

3.12.6.1 Overview

Ethernet Network Interface Controller (ENIC) enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2015.

3.12.6.2 Features

The device features one ENIC, supporting the following key features:

Standard compliance:

- MII/RMII/RGMII
- IEEE 802.3-2015 for Ethernet MAC
- IEEE 1588-2008 for precision networked clock synchronization
- IEEE 802.1AS-2011 and 802.1-Qav-2009 for Audio Video (AV) traffic
- IEEE 802.1Qbv-2015, 802.1Qbu-2016, and 802.1AS-Rev D5.0 for Time-Sensitive Networking (TSN) traffic
- IEEE 802.3az-2010 for Energy Efficient Ethernet (EEE)

MAC features:

- 10/100/1000 Mbps speed mode
- Half-duplex operation:
  - Support of CSMA/CD protocol
  - Support of flow control using backpressure
    - Full-duplex flow control operation (IEEE 802.3x pause packets and priority flow control)
    - Receive:
  - Automatic pad and CRC stripping options
  - Preamble and Start Frame Delimiter (SFD) deletion
  - Option to disable automatic CRC checking
  - Flexible address filtering modes:
    - Up to 31 additional 48-bit destination address filters with masks for each byte
    - Up to 31 x 48-bit source address comparison check with masks for each byte

        - 256-bit hash filter for multicast and unicast destination addresses
        - Option to pass all multicast addressed packets
        - Promiscuous mode to pass all packets without any filtering for network monitoring
    - Additional packet filtering:
        - VLAN tag-based: Perfect match and hash-based filtering. Filtering based on either outer or inner VLAN tag is possible.
        - Layer 3 and Layer 4-based: TCP or UDP over IPv4 or IPv6
        - Extended VLAN tag based with 4 filters selection
- Transmit:
    - Automatic pad and CRC generation control ability on a per-packet basis
    - Preamble and start of packet data insertion
    - Programmable packet length to support standard or jumbo Ethernet packets of up to 16 KB
    - Programmable inter packet gap (40-bit to 96-bit times in steps of 8)
    - IEEE 802.3x flow control automatic transmission of zero-quanta pause packet when the flow control input transitions from assertion to de-assertion (in full-duplex mode)
    - Source address field insertion or replacement, and VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control
    - Insertion, replacement, or deletion of up to two queue/channel-based VLAN tags

MAC Transaction Layer (MTL) features:

- Receive:
    - 16KB RX FIFO and 4 RX queues
- Transmit:
    - 16KB TX FIFO and 4 TX queues
    - Store-and-forward mechanism or threshold mode (cut-through) for transmission to the MAC
    - Calculate and insert IPv4 header checksum and TCP, UDP, or ICMP checksums
    - Scheduling algorithms:
    - Weighted Round Robin (WRR)
    - Strict Priority (SP)
    - Credit-Based Shaper (CBS) when audio-video bridging is enabled

Clause 22 and Clause 45 MDIO master interface for PHY configuration and management

3.12.6.3 Block Diagram

This section provides a block diagram to illustrate the main components and functions of the module. An AXI Master interface is connected to all DMA channels. The DMA arbiter provides helps in arbitration of all the paths (Transmit and Receive) in all channels. Each channel has a separate set of Control and Status registers (CSR) for managing the Transmit and Receive functions, descriptor handling, and interrupt handling.

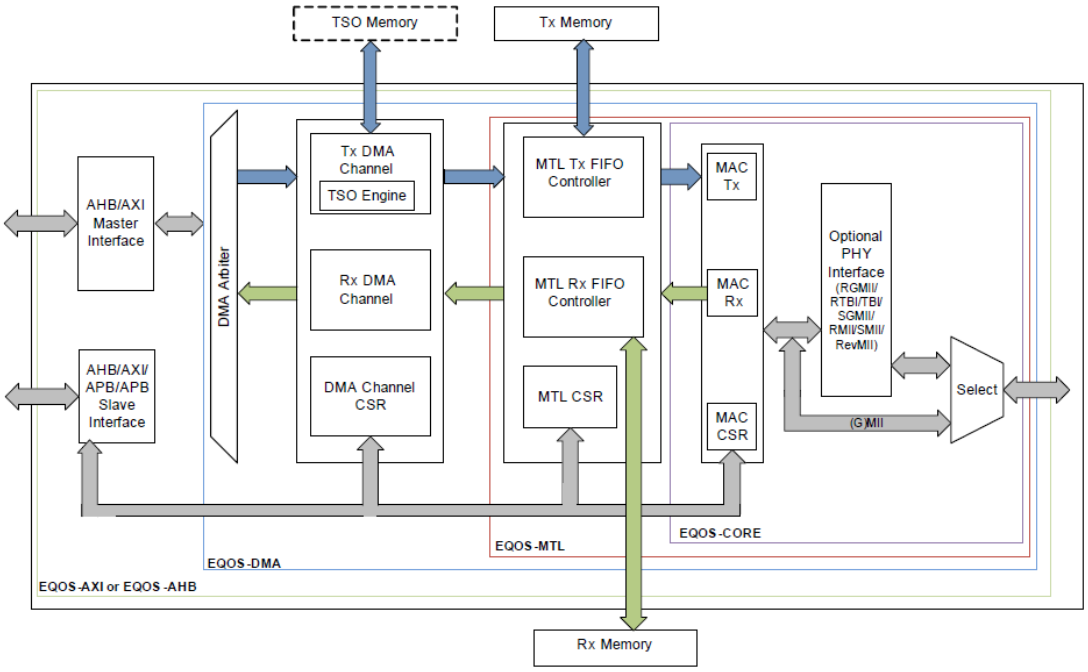


Figure 3-186 Block Diagram of ENIC

3.12.6.4 Function Description

3.12.6.4.1 Functions of IP

- AXI Master Interface
  The AXI is designed to integrate with AMBA AXI bus on the application side. The AXI transfers the data to and from the system memory through AXI master interface.

- APB Slave Interface
  The host CPU uses the APB slave interface to access the Control and Status registers (CSRs) of ETHER_QOS.

- PHY Interface
  ETHER_QOS supports the following PHY interfaces:
  - Reduced GMII (RGMII)
  - Media Independent Interface (MII)
  - Reduced MII (RMII)

- DMA
  The DMA has independent TX and RX engines, and a CSR space. The TX engine transfers data from the system memory to the device port (MTL), whereas the RX engine transfers data from the device port to the system memory. The DMA engine uses descriptors to efficiently move data from source to destination with minimal application CPU intervention. The DMA is designed for packet-oriented data transfers such as packets in Ethernet. The DMA controller can be programmed to interrupt the application CPU for situations such as Packet Transmit and Receive Transfer completion, and other normal or error conditions.

  The DMA and the application communicate through the following two data structures:

- Control and Status registers (CSR)
- Descriptor lists and data buffers

The DMA supports up to 4 TX and 4 RX Descriptor lists (or DMA channels). The base address of each list is written to the respective TX Descriptor List Address register and RX Descriptor List Address register. The descriptor list is forward linked and the next descriptor is always considered at a fixed offset to the current one. The offset is controlled by the DSL field of DMA_Ch[n]_Control register. The number of descriptors in the list is programmed in the respective TX (or RX) Descriptor Ring Length register. Once the DMA processes the last descriptor in the list, it automatically jumps back to the descriptor in the List Address register to create a descriptor ring. Note that the 4 TX and 4 RX queues can be controlled by multiple CPUs. The application should set access right protection in APMIXED.

The descriptor lists reside in the physical memory address space of the application. Each descriptor can point to a maximum of two buffers in the system memory. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.

A data buffer resides in the application physical memory space and consists of an entire packet or part of a packet but cannot exceed a single packet. Buffers only contain data. Buffer status is maintained in the descriptor. Data chaining refers to packets that span multiple data buffers. However, a single descriptor cannot span multiple packets. The DMA skips to the data buffer of the next packet when EOP is detected.

- Transaction Layer (MTL)
  The MAC Transaction Layer (MTL) provides the FIFO memory Interface to buffer and regulate the packets between the application system memory and the MAC. It also enables the data to be transferred between the application clock and MAC clock domains. The MTL layer has two data paths: Transmit path and Receive Path.

- MAC
  The MAC supports RGMII/MII/RMII towards the PHY chip. The PHY interface can be selected only once after reset.

3.12.6.5 Theory of Operations

3.12.6.5.1 Ethernet Frame Formats

The IEEE 802.3 standard defines the Ethernet frame format as follows: An Ethernet frame has a minimum length of 64 bytes and a maximum length of 1518 bytes, excluding the preamble and the SFD bytes. An Ethernet frame consists of the following fields:

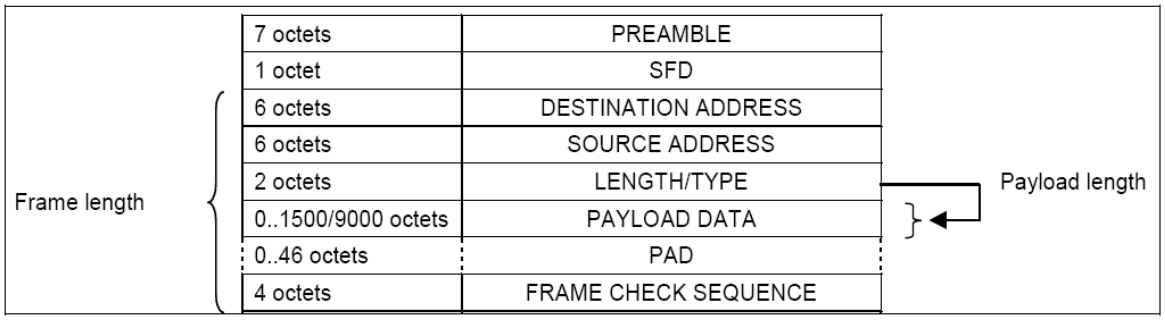


Figure 3-187 Ethernet Frame Format without VLAN Tag

Optional MAC frames can be VLAN tagged with an additional 4-bytes field (VLAN tag and VLAM info) inserted between the MAC Source Address and the Length/Type Field. VLAN tagging is defined by the IEEE P802.1q specification.

VLAN Tagged Ethernet Frame Format is as follows.

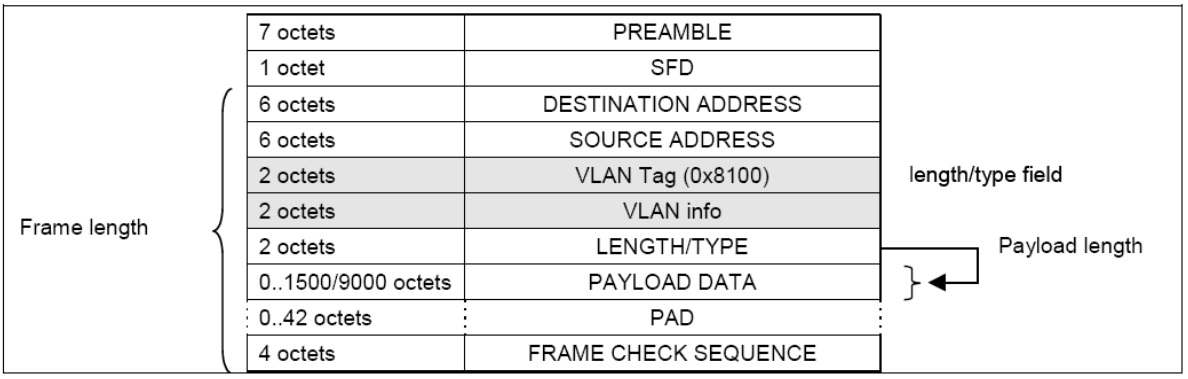


Figure 3-188 Ethernet Frame Format with VLAN Tag

3.12.6.5.2 Pause Frames

The IEEE 802.3 defined pause frame has the following format:

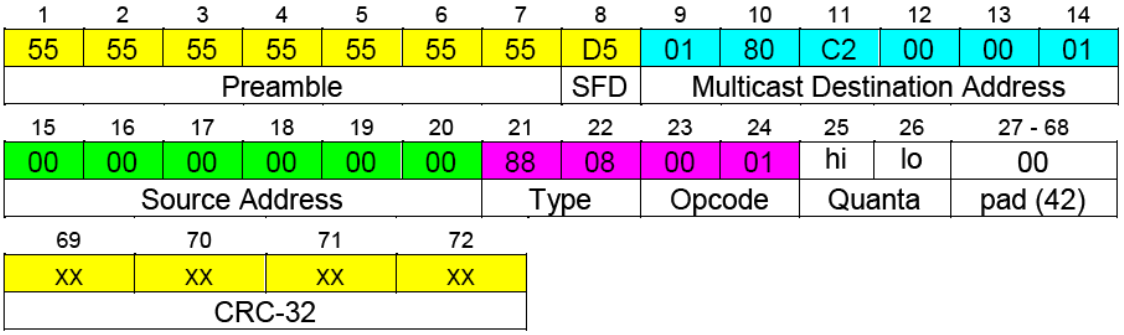


Figure 3-189 Pause Frame Format

There is no Payload Length field found within a Pause Frame and a Pause Frame is always padded with 42 bytes (0x00). If a pause frame with a pause value greater than zero (XOFF Condition) is received, the MAC stops transmitting data as soon as the current Frame transfer is completed. The MAC stops transmitting data for the value defined in pause quanta. One pause quanta fraction refers to 512 bit times.

If a pause frame with a pause value of zero (XON Condition) is received, the transmitter is allowed to send data immediately.

3.12.6.5.3 Line Interfaces

The Core implements a MII for 10/100 Mbps, a RMII for 10/100 Mbps and a RGMII for 10/100/1000 Mbps mode of operation.

3.12.6.5.4 MII for 10M/100M MAC

On Transmit, all data transfers are synchronous to `tx_clk` rising edge. The MII data enable signal `txen_o` is asserted to indicate the start of a new frame and remains asserted until the last byte of the frame is present on `txd_o(3:0)` bus. Between frames, `txen_o` remains de-asserted.

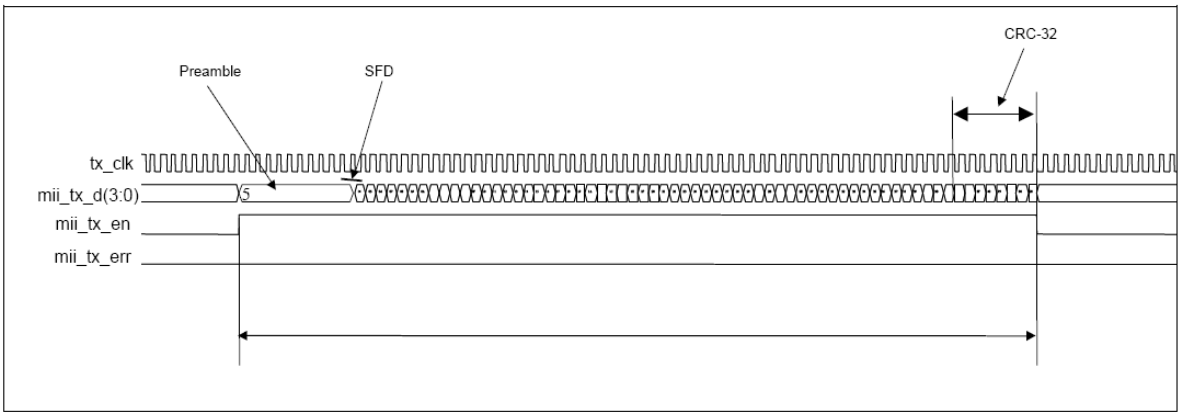


Figure 3-190 MII Transmit Waveform

If a frame experiences internal errors the frame is subsequently transmitted with the MII `mii_txer` error signal for one clock cycle at any time during the packet transfer.

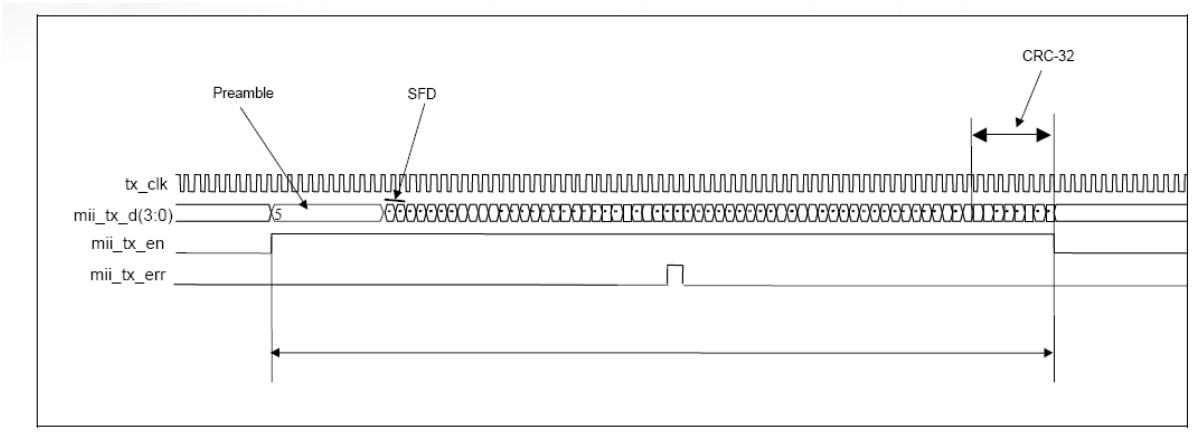


Figure 3-191 MII Transmit Waveform with Error Occurring

On receive, all signals are sampled on the `rx_clk` rising edge. The MII data enable signal `rxdv_i` is asserted by the PHY to indicate the start of a new frame and remains asserted until the last byte of the frame is present on `rxd_i(3:0)` bus. Between frames, `rxdv_i` remains de-asserted.

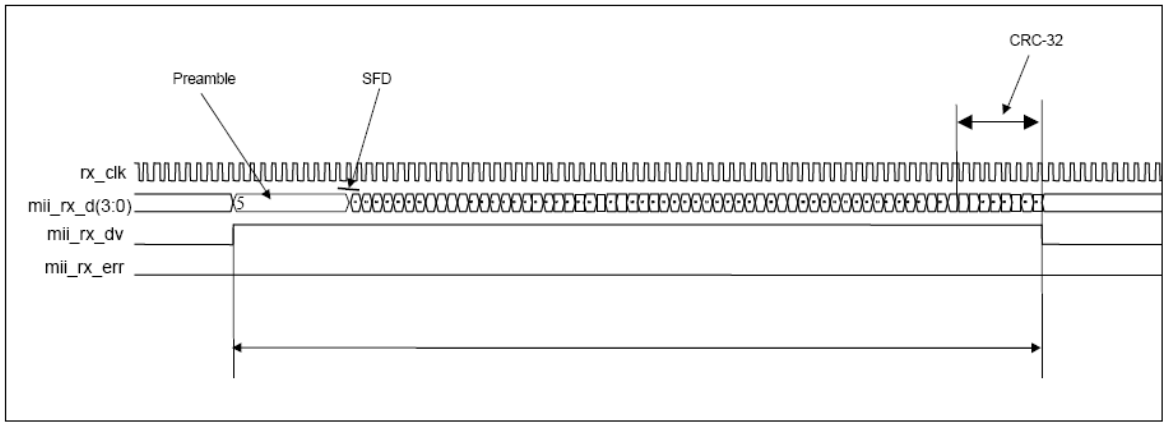


Figure 3-192 MII Receive Waveform

If the PHY detects an error on the frame received from the line, the PHY asserts the MII error signal, `rxer_i`, for at least one clock cycle at any time during the packet transfer.

3.12.6.5.5 RGMII for 10M/100M/1000M

On Transmit for 1000M, all data transfers are synchronous to `tx_clk` rising edge or `tx_clk` falling edge. The GMII data enable signal `txen_o` is asserted to indicate the start of a new frame and remains asserted until the last byte of the frame is present on `txd_o(3:0)` bus. Between frames, `txen_o` remains de-asserted.

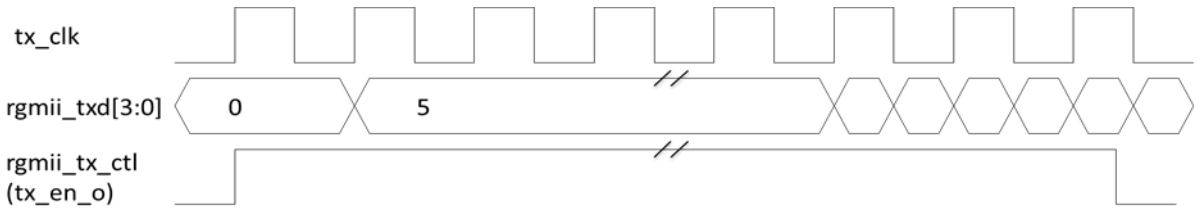


Figure 3-193 RGMII Transmit Waveform in 1000M

If a frame experiences internal errors, the frame is subsequently transmitted with `txen_o` de-asserting synchronous to `tx_clk` rising edge or `tx_clk` falling edge at any time during the packet transfer.

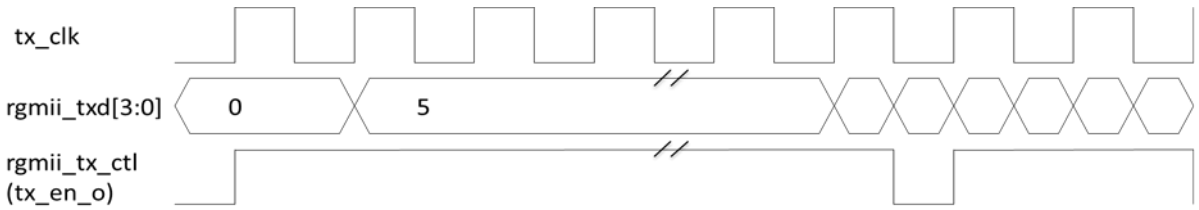


Figure 3-194 RGMII Transmit Waveform with Error Occurring in 1000M

On receive for 1000M, all signals are sampled on `rx_clk` rising edge or `rx_clk` falling edge. The RGMII data enable signal `rxdv_i` is asserted by the PHY to indicate the start of a new frame and remains asserted until the last byte of the frame is present on `rxd_i(3:0)` bus. Between frames, `rxdv_i` remains de-asserted.

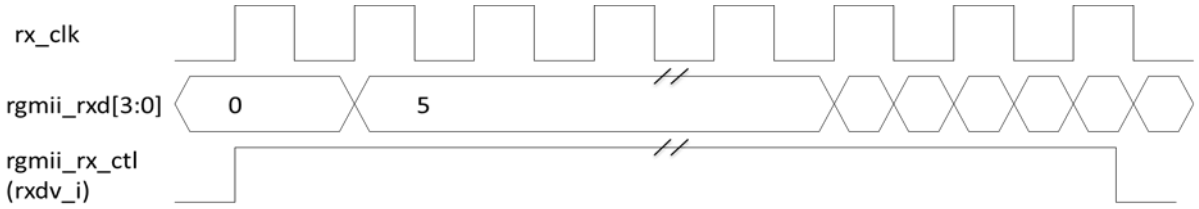


Figure 3-195 RGMII Receive Waveform in 1000M

If the PHY detects an error on the frame received from the line, the PHY asserts the RGMII error signal, `rxdv_i`, for at least half clock cycle at any time during the packet transfer.

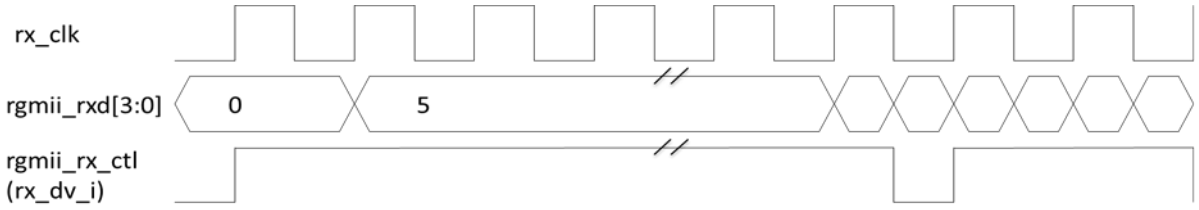


Figure 3-196 RGMII Receive Waveform with Error Occurring in 1000M

On Transmit for 10/100M, all data transfers are synchronous to `tx_clk` rising edge. The GMII data enable signal `txen_o` is asserted to indicate the start of a new frame and remains asserted until the last byte of the frame is present on `txd_o(3:0)` bus. Between frames, `txen_o` remains de-asserted. Note that the frequency of tx_clk is 25M in 100M mode and 2.5M in 10M mode.

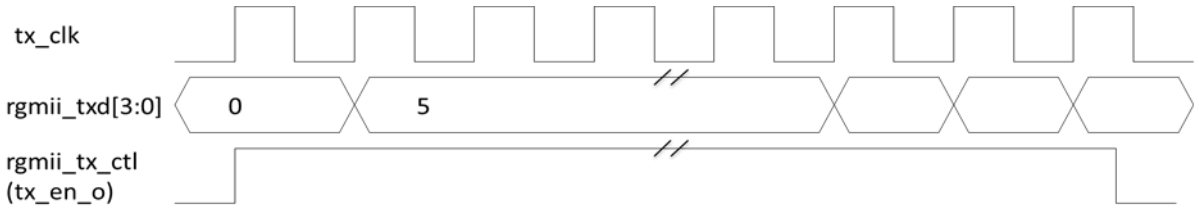


Figure 3-197 RGMII Transmit Waveform in 10/100M

If a frame experiences internal errors, the frame is subsequently transmitted with `txen_o` de-asserting synchronous to `tx_clk` rising edge or `tx_clk` falling edge at any time during the packet transfer.

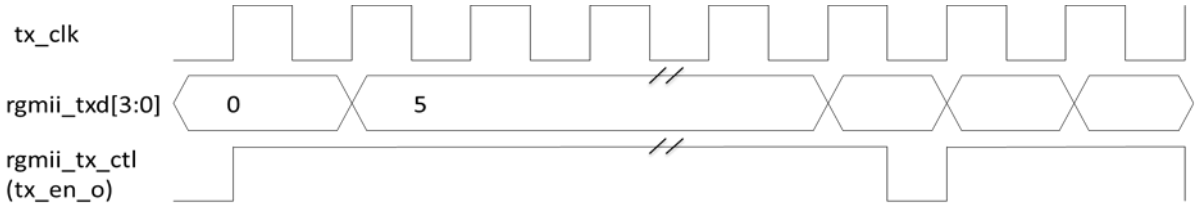


Figure 3-198 RGMII Transmit Waveform with Error Occurring in 10/100M

On receive for 10/100M, all signals are sampled on `rx_clk` rising edge. The RGMII data enable signal `rxdv_i` is asserted by the PHY to indicate the start of a new frame and remains asserted until the last byte of the frame is present on `rxd_i(3:0)` bus. Between frames, `rxdv_i` remains de-asserted. Note that the frequency of rx_clk is 25M in 100M mode and 2.5M in 10M mode.

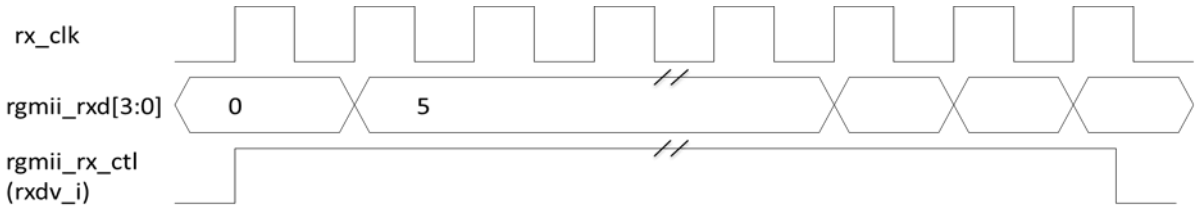


Figure 3-199 RGMII Receive Waveform in 10/100M

If the PHY detects an error on the frame received from the line, the PHY asserts the RGMII error signal, `rxdv_i`, for at least half clock cycle at any time during the packet transfer.

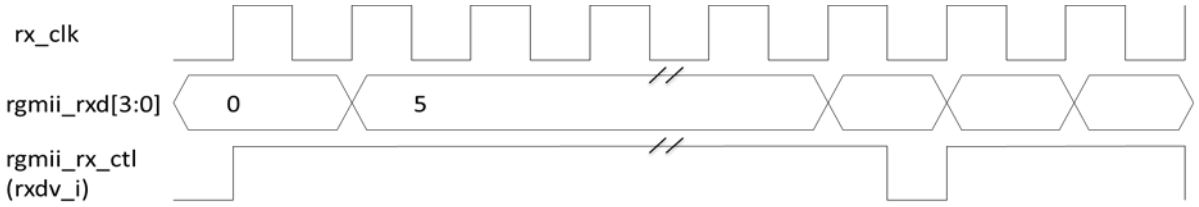


Figure 3-200 RGMII Receive Waveform with Error Occurring in 10/100M

3.12.6.5.6 RMII for 10/100 MAC

The RMII specification reduces the data interfaces from 4-bit (nibble) data to 2-bit (di-bit) data. In addition, control is reduced to 3 signals and one clock. Thus, the total signal connection is reduced to 8 pins.

The following figure shows the RMII mode connection between a MAC and a RMII Ethernet Transceiver.

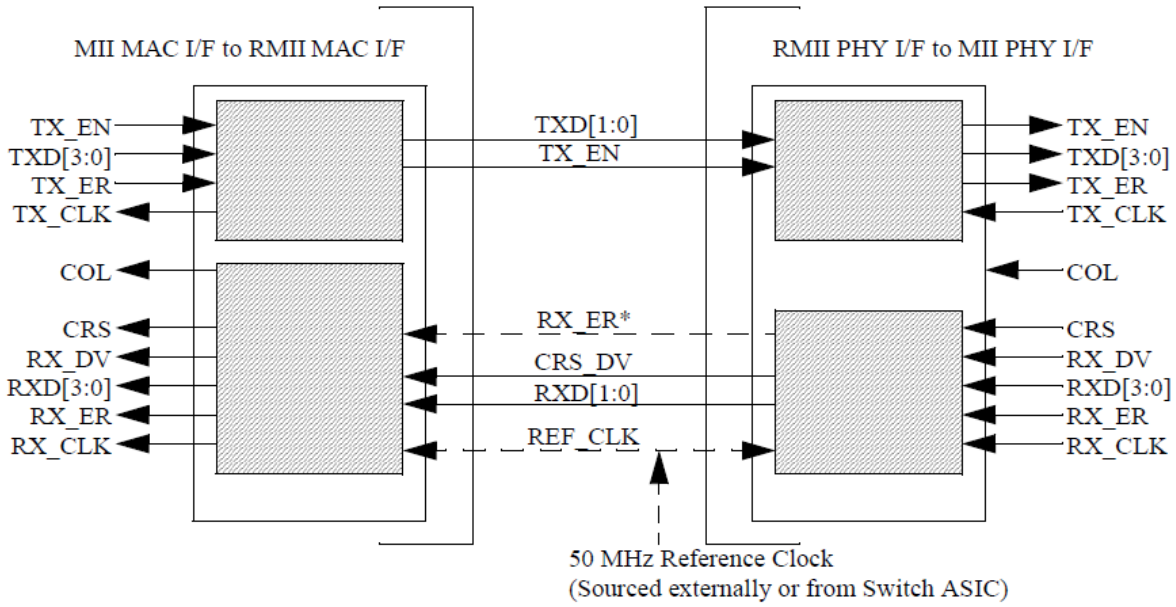


Figure 3-201 RMII Signal Connection between PHY and MAC

- TX signals:
  TX_EN indicates that the MAC is presenting di-bits on TXD[1:0] for transmission. TX_EN shall be asserted synchronously with the first nibble of the preamble and shall remain asserted while all di-bits to be transmitted are

presented. The MAC should assert TX_EN negated prior to the first REF_CLK rising edge following the final di-bit of a frame.

TX_EN shall transition synchronously with respect to REF_CLK.

TXD[1:0] shall transition synchronously with respect to REF_CLK. When TX_EN is asserted, TXD[1:0] are accepted for transmission by the PHY.

In 10Mbps operation, as the REF_CLK frequency is 10 times the data rate in 10Mb/s mode, the value on TXD[1:0] must be stable for 10 clocks, allowing the PHY to sample every 10th cycle.

- RX signals:
  CRS_DV shall be asserted by the PHY when the receive medium is non-idle. CRS_DV is asserted asynchronously on detection of carrier due to the criteria relevant to the operating mode.

  The data on RXD[1:0] is considered valid once CRS_DV is asserted. However, since the assertion of CRS_DV is asynchronous relative to REF_CLK, the data on RXD[1:0] shall be “00” until proper receive signal decoding takes place.

The following figure shows CRS_DV formation.

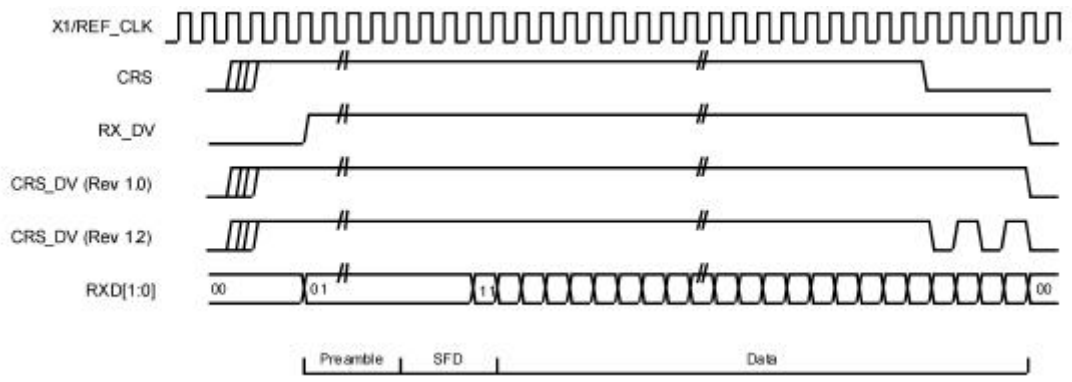


Figure 3-202 RMII Receive Waveform

As the REF_CLK frequency is 10 times the data rate in 10 Mbps mode, the value on RXD[1:0] may be sampled every 10th cycle by the MAC.

3.12.6.5.7 MAC Receive

The MAC receive engine performs the following tasks:

- Check Frame Framing
- Remove Frame preamble and Frame SFD field
- Terminate Pause Frames
- Support 9 (8 of them are optional) configurable DA to check received DA. If not matching, the packet will be dropped.
- Drop processing of oversized frame and short frame.
- Calculate and verify CRC-32
- Write received Frames in the Core receive FIFO
- IP and TCP/UDP checksum

- DMA to write packet data from receive FIFO to external memory.
- DMA interface transfer to AXI

1. Preamble Processing

   MAC Core checks for the start frame delimiter (SFD) byte. Before SFD, 0 to 7 bytes preamble is acceptable. The following shows cases of no preamble and odd preamble.

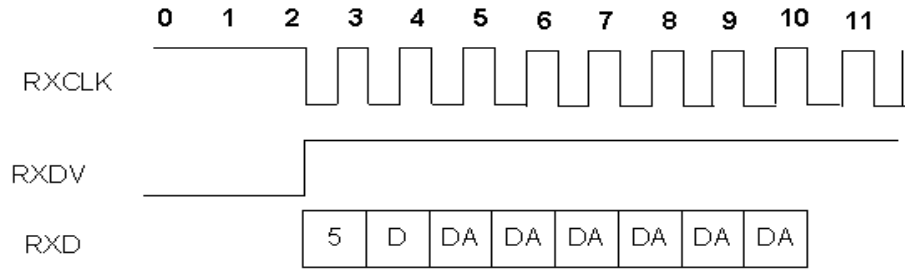


Figure 3-203 No Preamble Case

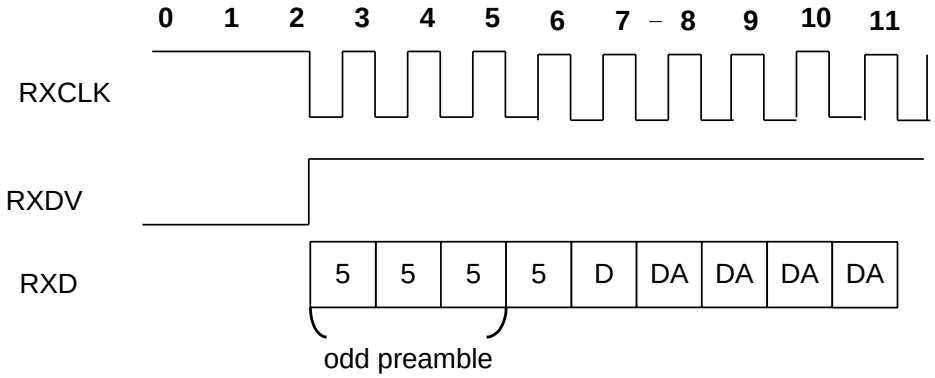


Figure 3-204 Odd Preamble Case

2. Frame Length / Type Verification

   The NIC does not check the correction of length field. An internal counter is used to calculate frame length. If calculated length < 64 bytes, then drop as a runt packet. If calculated length > 1518 (or 1522, 1536, by configuration), then assert oversize indication in frame information and this packet will be optionally dropped in DMA.

   Control and VLAN frames (Frame Length/Type field 0x8808 and 0x8100 respectively) are processed by the Core as described in the two following sections.

   The NIC supports 802.1q tag-based VLAN ingress check, and it can support up to 4 VLANs, set in registers, where these VLAN IDs can be any in 4K VLAN space. Internally, the controller uses 4 bits of "My VLAN ID Control Register" to enable VLAN ingress check for each pre-defined VLAN ID. When at least one of the pre-defined VLAN ID is enabled, RX MAC will compare the pre-defined VLAN ID with the tagged VID of the received packet. If one of them is matched, the packet will be received; otherwise, it will be dropped, and the relevant MIB counter will be increased by 1 accordingly.

   Please note that VLAN ingress check has no effect on non-VALN tagged packet. When a received packet is VLAN-tagged, the tag can be stripped from the packet or retained with the packet. No matter VLAN tag is stripped or not, the VLAN tag information will be stamped at RX Descriptor.

3. Pause Frame Processing

   Pause frame are not transferred to the receive FIFO.

   A pause frame is valid only, if all the following conditions are valid:

   - Length/Type is set to 0x8808
   - The Opcode immediately following the Length/Type field is 0x0001
   - The frame MAC destination address is either the configured unicast address (Registers MAC_ADDR_0 and MAC_ADDR_1) or the control frame multicast address 01-80-c2-00-00-01
   - The frame has a valid CRC
   - The frame has a length of 64 octets

   Check if the received packet is a pause frame and generate signal to pause TX (pause_tx) when receiving pause on frame.

4. CRC

   The CRC-32 field is always checked in the received side. The CRC polynomial, as specified in the 802.3 Standard, is:

   $FCS(X) = X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X^{1}+1$

   The 32 bits of the CRC value are placed in the FCS field so that the X31 term is the right-most bit of the first octet. The CRC bits are thus received in the following order: X31, X30,..., X1, X0.

   If a CRC-32 error is detected, the frame is marked invalid and the frame status bit 1 indicating a CRC error is set to "1".

5. Frame Padding

   In receive, the MAC does not remove the padding octets even if the Payload length is less than 46 bytes (42 bytes for VLAN tagged frames)

6. Frame Truncation

   Since NIC does not do length field checking, so that function of frame truncation is not implemented.

7. Hash Table

   A 256-bit Hash table is implemented for multicast and unicast addresses filter function. It operates as follows.

   First, on receiving a multicast MAC address frame, it calculates 8-bit Hash Value from destination MAC address of the frame.

   Next, it compares the calculated Hash Value with the SelectBit_n(n = 0 to 255) references of the MAC Address Hash Filter Table.

   Last, it decides whether to forward the frame to memory or to deny it.

   If a Hash Value of RX frame MAC address becomes n and the SelectBit_n is 0, the RX frame is denied. Otherwise, if the SelectBit_n is 1, the RX frame is forwarded to memory.

The Hash Value results is aggregated form CRC32 calculation from when CRC32 is used in the following generator polynomial to degenerate the destination MAC address (48 bits).

Generator Polynomial: = $X_{32}+X_{26}+X_{23}+X_{22}+X_{16}+X_{12}+X_{11}+X_{10}+X_{8}+X_{7}+X_{5}+X_{4}+X_{2}+X_{1}+1$

The 8-bit hash value can be generated from reversed CRC

{r_crc[31],r_crc[30],r_crc[29],r_crc[28],r_crc[27],r_crc[2],r_crc[1],r_crc[0]}

The following shows a 256-bit hash table and the process of hash value generation.

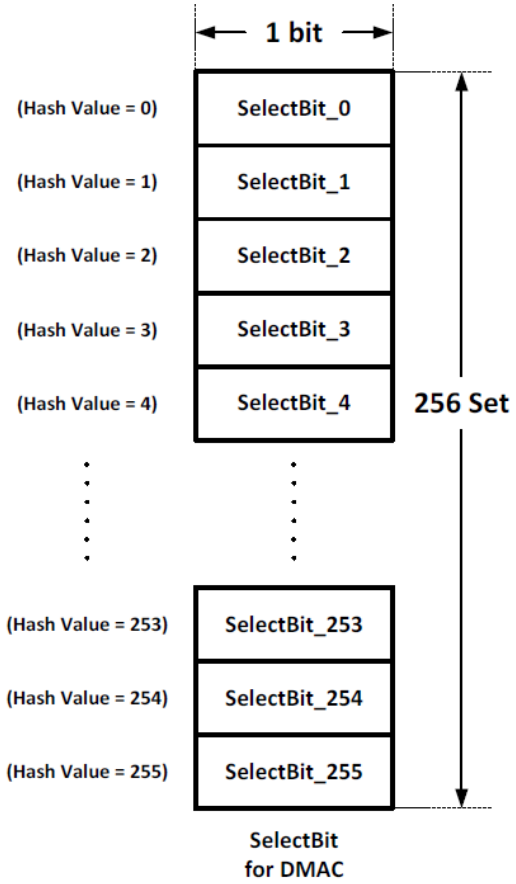


Figure 3-205 Hash Table Set

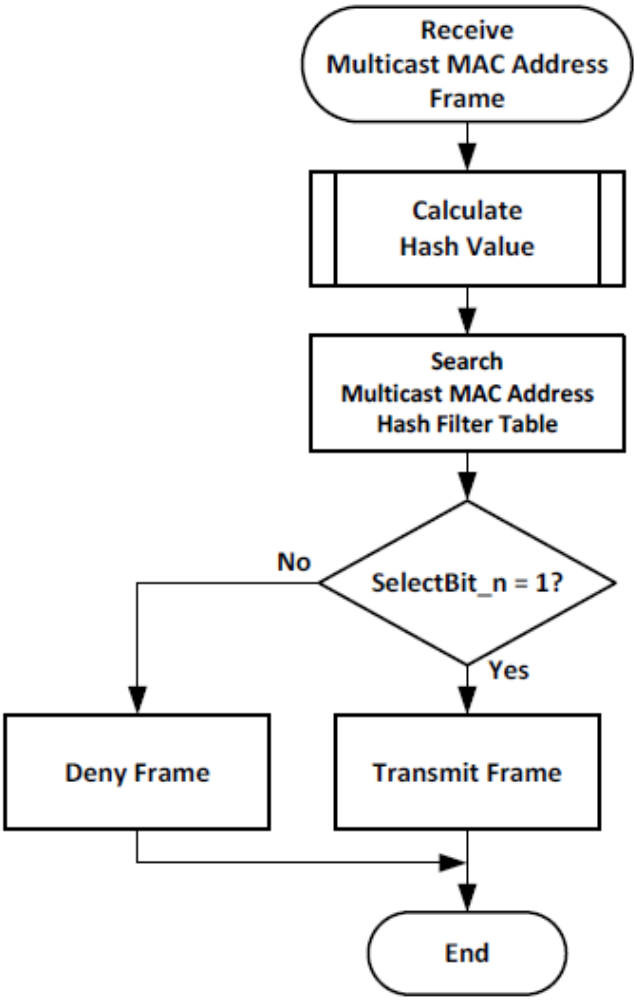


Figure 3-206 Hash Table Control Flow Chart

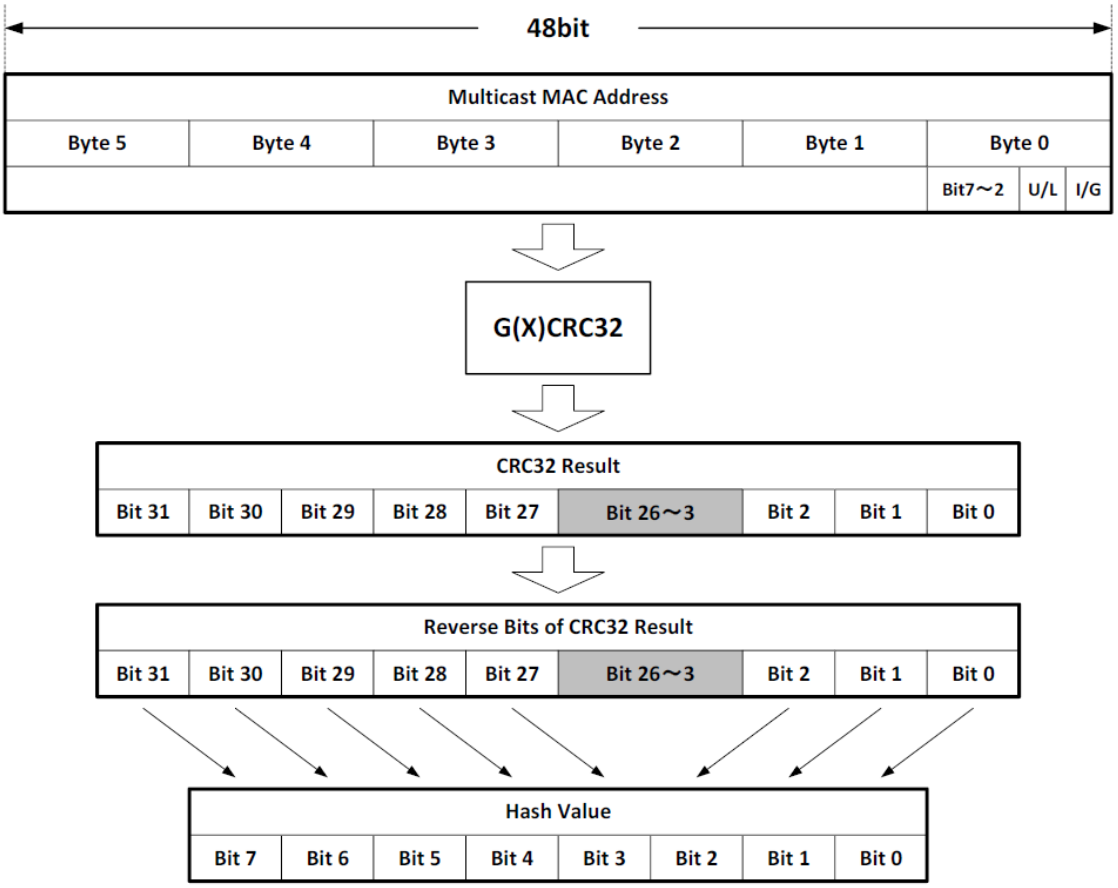


Figure 3-207 Hash Value Calculation

8. Magic Packet Detection
   Wake-on-LAN ("WOL") is implemented using a specially designed packet called a magic packet.

   The magic packet is a broadcast frame containing anywhere within its payload 6 bytes of all 255 (FF FF FF FF FF FF in hexadecimal), followed by sixteen repetitions of the target computer's 48-bit MAC address, for a total of 102 bytes.

   When Wake-on-LAN is enabled, TX MAC will be powered down and RX MAC will only scan Magic Packet and not forward any packet to system memory. After detecting the Magic Packet, MAC asserts WOL interrupt to CPU and wake-up CPU accordingly.

3.12.6.5.8 MAC Transmit

Ethernet Frame transmission starts when the Transmit FIFO holds enough data. Once a transfer has started, the transmit engine performs the following tasks:

- Convert word to byte
- Generate Preamble and SFD field before Frame transmission
- When in Link Pause Mode, generate Pause frames if the Receive FIFO reports a congestion or if the pause generation pin back_pressure on gtx_main is asserted
- When in Link Pause Mode, suspend Ethernet Frame transfer (XOFF) if a non-zero Pause Quanta is received from the MAC receive path (optional)
- Calculate and replace CRC-32 to the transmitted frame (optional)

- Send Frame with correct Inter Packet Gap (IPG)

1G/100M/10M mode default stop TX when link status of PHY is deasserted.
The following figure shows the TX transmit flow chart.

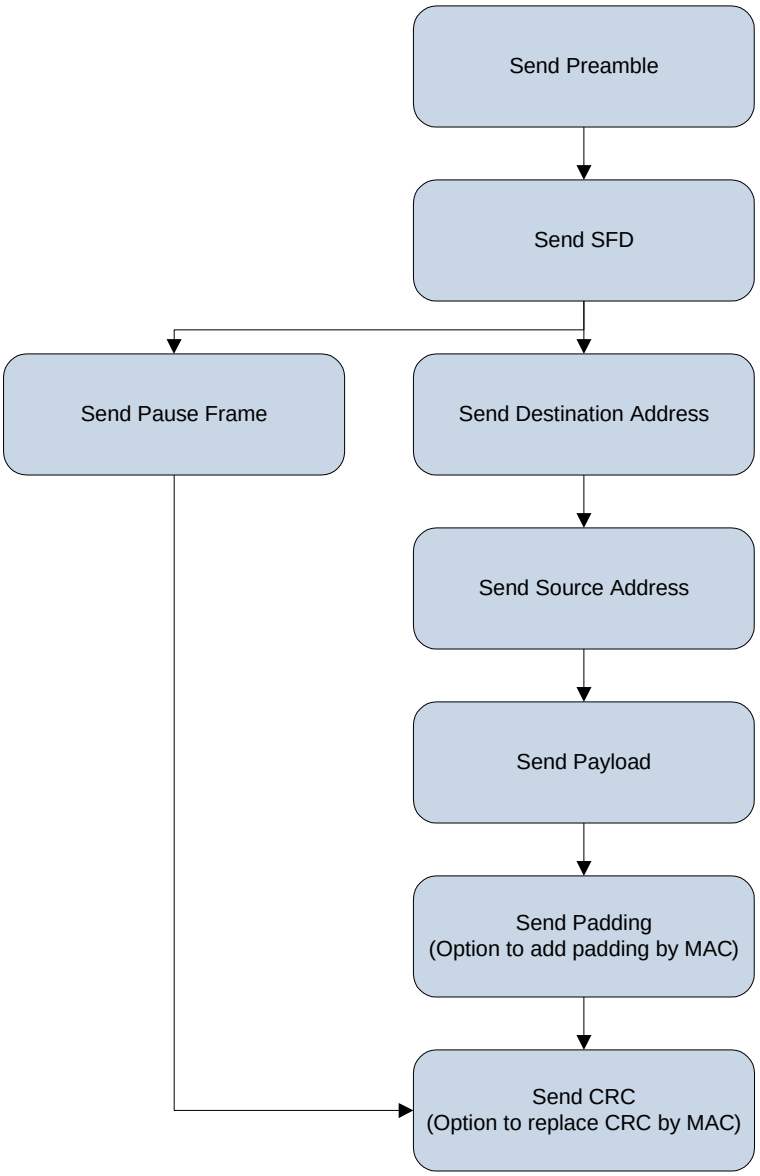


Figure 3-208 Transmit Flow Chart

3.12.6.6 Signal Descriptions

Table 3-143 presents ENIC signal descriptions.

Table 3-143 ENIC Signal Descriptions

Signal name	Type	Description	Ball location
ENIC Receive Data Bus—GBE_RXD[3:0]			
GBE_RXD0	DI	Gigabit Ethernet RX data 0	AK10
GBE_RXD1	DI	Gigabit Ethernet RX data 1	AK11
GBE_RXD2	DI	Gigabit Ethernet RX data 2	AR14
GBE_RXD3	DI	Gigabit Ethernet RX data 3	AU15

Signal name	Type	Description	Ball location
ENIC Transmit Data Bus—GBE_TXD[3:0]			
GBE_TXD0	DO	Gigabit Ethernet TX data 0	AK14
GBE_TXD1	DO	Gigabit Ethernet TX data 1	AL14
GBE_TXD2	DO	Gigabit Ethernet TX data 2	AM14
GBE_TXD3	DO	Gigabit Ethernet TX data 3	AN14
ENIC Command, Status, Clock and Interrupt Signals			
GBE_COL	DI	Gigabit Ethernet collision detected	AU14
GBE_INTR	DI	Gigabit Ethernet interrupt from external PHY	AT13
GBE_RXC	DI	Gigabit Ethernet RX clock	AR12
GBE_RXDV	DI	Gigabit Ethernet RX data valid	AL12
GBE_RXER	DI	Gigabit Ethernet RX error	AN12
GBE_TXC	DIO	Gigabit Ethernet TX clock	AP14
GBE_TXEN	DO	Gigabit Ethernet TX data valid	AK12
GBE_TXER	DO	Gigabit Ethernet TX error	AP13
ENIC Management Bus			
GBE_MDC	DO	Gigabit Ethernet MDC	AP12
GBE_MDIO	DIO	Gigabit Ethernet MDIO	AL13

3.12.6.7 ENIC Timing Characteristics

Table 3-144 and Figure 3-209 present timing characteristics for the ENIC MII in the device.

Table 3-144 ENIC MII Timing Characteristics

No.	Parameter		Min	Typ	Max	Unit
MII1	$t_{c\_TXC\_RXC}$	Cycle time, TXC/RXC		40		ns
MII2	t_{d_TX}	Delay time, transmission			9.764	ns
MII3	t_{d_TXC}	Delay time, TXC	1.323		2.273	ns
MII4	t_{d_TXD}	Delay time, TXD/TXEN/TXER	4.149		7.491	ns
MII5	t_{d_RXC}	Delay time, RXC	1.514		2.654	ns
MII6	t_{d_RXD}	Delay time, RXD/RXDV/RXER	2.041		4.021	ns
MII7	t_{su_RX}	Setup time, RXD/RXDV/RXER	3.507			ns
MII8	t_{h_RX}	Hold time, RXD/RXDV/RXER	0.473			ns
	D	Duty cycle, TXC/RXC	40	50	60	%
MII10	t_{RISE}	Rise time, TXC/RXC (20% ~ 80%)			0.75	ns
MII11	t_{FALL}	Fall time, TXC/RXC (20% ~ 80%)			0.75	ns

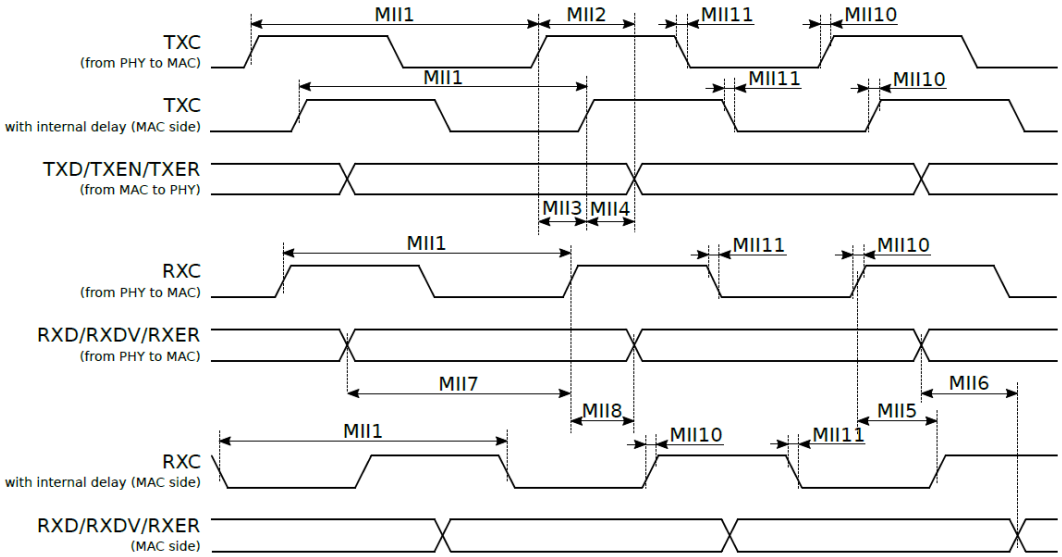


Figure 3-209 ENIC MII Timing Diagram

Table 3-145 and Figure 3-210 ENIC RMII Timing Diagram present timing characteristics for the ENIC RMII in the device.

Table 3-145 ENIC RMII Timing Characteristics

No.	Parameter		Min	Typ	Max	Unit
RMII1	t_{c_REFCLK}	Cycle time, REFCLK		20		ns
RMII2	t_{d_TX}	Delay time, transmission			7.235	ns
RMII3	t_{d_TXC}	Delay time, REFCLK (when transmitting)	1.144		2.2	ns
RMII4	t_{d_TXD}	Delay time, TXD/TXEN	2.8		5.035	ns
RMII5	t_{d_RXC}	Delay time, REFCLK (when receiving)	1.798		3.071	ns
RMII6	t_{d_RXD}	Delay time, RXD/RXDV	1.925		3.469	ns
RMII7	t_{su_RX}	Setup time, RXD/RXDV	2.671			ns
RMII8	t_{h_RX}	Hold time, RXD/RXDV	0.873			ns
	D	Duty cycle, REFCLK	45	50	55	%
RMII10	t_{RISE}	Rise time, REFCLK (20%~80%)			0.75	ns
RMII11	t_{FALL}	Fall time, REFCLK (20%~80%)			0.75	ns

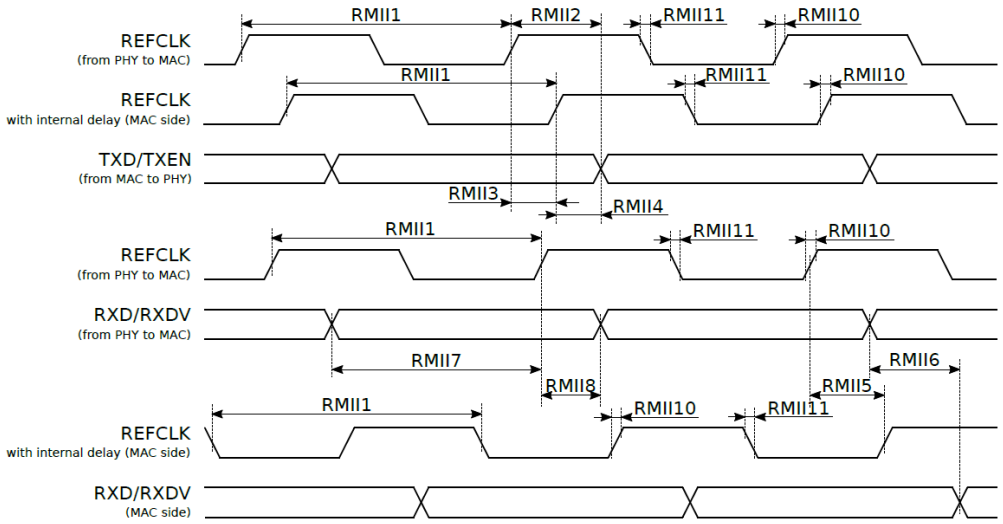


Figure 3-210 ENIC RMII Timing Diagram

Table 3-146 and Figure 3-211 ENIC RGMII Timing Diagram present timing characteristics for the ENIC RGMII in the device.

Table 3-146 ENIC RGMII Timing Characteristics

No.	Parameter		Min	Typ	Max	Unit
RGMII1	t_{TXC_RXC}	Cycle time, TXC/RXC (10M/100M/1000M)		400/40/8		ns
RGMII2	t_{d_TX}	Delay time, transmission			4.1 + K(1)	ns
RGMII3	t_{d_TXC}	Delay time, TXC	3.489		8.169	ns
RGMII4	t_{d_TXD}	Delay time, TXD/TX_CTL	4.141		7.491	ns
RGMII5	t_{d_RXC}	Delay time, RXC	1.998		3.575	ns
RGMII6	t_{d_RXD}	Delay time, RXD/RX_CTL	1.913		3.548	ns
RGMII7	t_{su_RX}	Setup time, RXD/RX_CTL	2.55 - Q(2)			ns
RGMII8	t_{h_RX}	Hold time, RXD/RX_CTL	1.085 + Q(2)			ns
	D	Duty cycle, TXC/RXC (1000M)	45	50	55	%
		Duty cycle, TXC/RXC (10M/100M)	40	50	60	%
RGMII10	t_{RISE}	Rise time, TXC/RXC (20%~80%)			0.75	ns
RGMII11	t_{FALL}	Fall time, TXC/RXC (20%~80%)			0.75	ns

(1) K is programmable MAC internal delay (32-levels). The maximum delay is from 12.6001 ns to 28.5724 ns.

(2) Q is programmable MAC internal delay (32-levels). The maximum delay is from 12.6001 ns to 28.5724 ns.

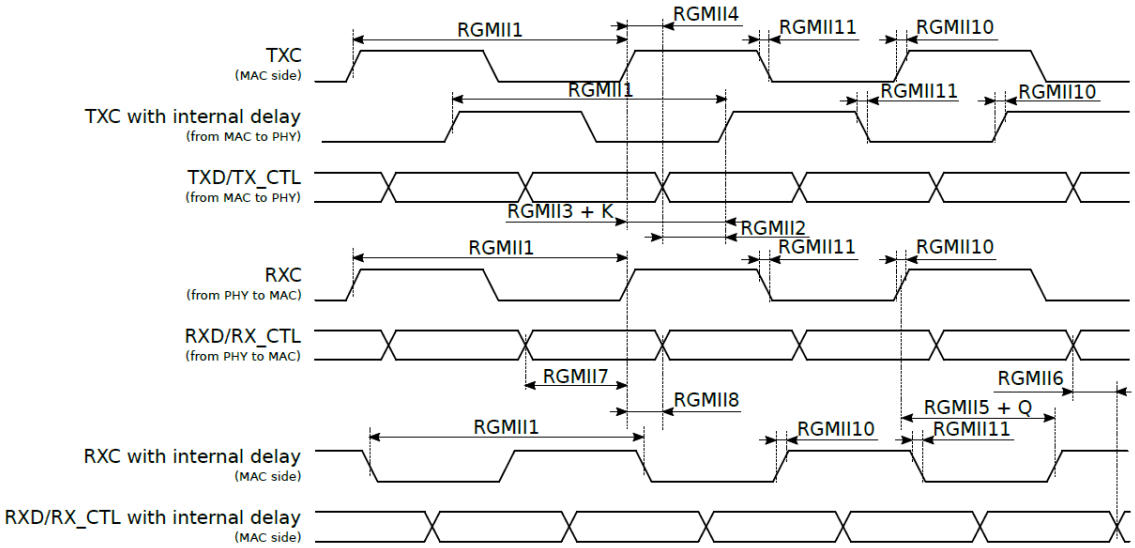


Figure 3-211 ENIC RGMII Timing Diagram

Table 3-147 and Figure 3-212 present timing characteristics for the ENIC MDIO in the device.

Table 3-147 ENIC MDIO Timing Characteristics

No.	Parameter		Min	Typ	Max	Unit
MDIO1	t_{c_MDC}	Cycle time, MDC	400			ns
MDIO2	t_{d_MDO}	Delay time, MDIO output		MDIO4 - MDIO3		ns
MDIO3	t_{d_MDC}	Delay time, MDC	3.631		9.043	ns
MDIO4	$t_{d\_MDO\_MAC}$	Delay time, MDIO output (MAC to PHY)	4.804		11.479	ns
MDIO5	t_{d_MDI}	Delay time, MDIO input	1.92		4.203	ns
MDIO6	t_{su_MDI}	Setup time, MDIO input	1 + MDIO5 + MDIO3			ns
MDIO7	t_{h_MDI}	Hold time, MDIO input	1 - MDIO5 - MDIO3			ns

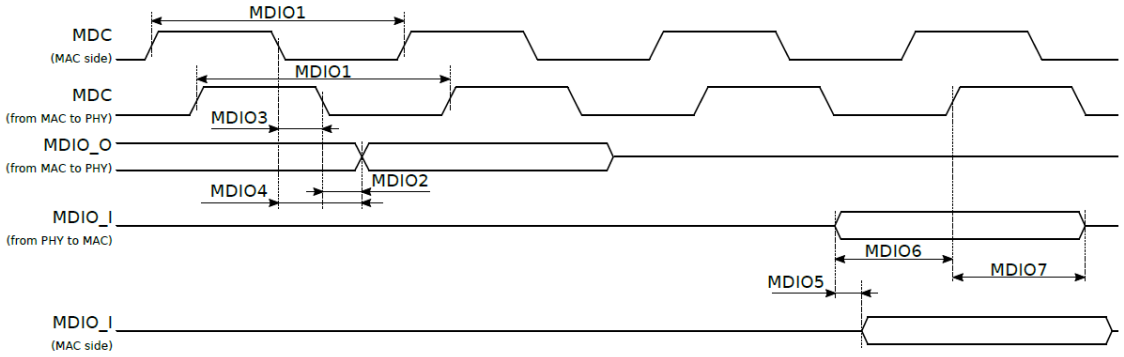


Figure 3-212 ENIC MDIO Timing Diagram

3.12.6.8 Programming Guide

- Initializing DMA
  Complete the following steps to initialize the DMA:

  1. Provide a software reset. This resets all of the MAC internal registers and logic. (bit 0 of DMA_Mode).
  2. Wait for the completion of the reset process (poll bit 0 of the DMA_Mode, which is only cleared after the reset operation is completed).
  3. Program the following fields to initialize the DMA_SysBus_Mode Register:
     a. AAL
     b. Fixed burst or undefined burst
     c. Burst mode values in the case of AHB interface, OSR_LMT in the case of AXI bus interface.
     d. If fixed length value is enabled, select the maximum burst length possible on the AXI bus (bits [7:1])
  4. Create a descriptor list for TX and RX. In addition, ensure that the RX descriptors are owned by DMA (set bit 31 of descriptor TDES3/RDES3).
  5. Program the TX and RX Ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). The ring length must be programmed to at least 4.
  6. Initialize RX and TX descriptor list addresses with the base address of the TX and RX descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). Also, program the TX and RX tail pointer registers indicating to the DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)).
  7. Program the settings of the following registers for the parameters like maximum burst-length (PBL) initiated by DMA, descriptor skip lengths, OSP in the case of TxDMA, RBSZ in the case of RxDMA, and so on:
     - DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
     - DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
     - DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)
  8. Enable the interrupts by programming the DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.
  9. Start the RX and TX DMAs by setting SR (bit 0) of the DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) and ST (bit 0) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to 10. Repeat steps 4 to 9 for all the TX DMA and RX DMA selected channels in the hardware.
  10. Repeat steps 4 to 9 for all the TX DMA and RX DMA channels selected in the hardware.

- Initializing MTL Registers
  Complete the following steps to initialize the MTL Registers:

  1. Program the TX Scheduling (SCHALG) and Receive Arbitration Algorithm (RAA) fields in MTL_Operation_Mode to initialize the MTL operation in the case of multiple TX and RX queues.
  2. Program the RX Queue to DMA mapping in MTL_RxQ_DMA_Map0 and MTL_RxQ_DMA_Map1 registers.
  3. Program the following fields to initialize the mode of operation in the MTL_TxQ0_Operation_Mode register
     a. Transmit Store And Forward (TSF) or Transmit Threshold Control (TTC) in the case of threshold mode
     b. Transmit Queue Enable (TXQEN) to value 2'b10 to enable Transmit Queue0
     c. Transmit Queue Size (TQS)

4. Program the following fields to initialize the mode of operation in the MTL_RxQ0_Operation_Mode register:
   a. Receive Store and Forward (RSF) or RTC in the case of Threshold mode
   b. Flow Control Activation and De-activation thresholds for MTL Receive FIFO (RFA and RFD)
   c. Enable Error Packet and undersized good Packet forwarding (FEP and FUP)
   d. Receive Queue Size (RQS)
5. Repeat previous two steps for all MTL TX and RX queues selected in the configuration

- Initializing MAC

  The following MAC Initialization operations can be performed after DMA initialization. If the MAC initialization is completed before the DMA is configured, enable the MAC RX (last step in the following sequence) only after the DMA is active. Otherwise, received frames will fill in the RX FIFO and overflow.

  1. Provide the MAC address registers: MAC_Address0_High and MAC_Address0_Low. If more than one MAC address is enabled in your configuration (during configuration in the coreConsultant), program the MAC addresses appropriately.
  2. Program the following fields to set the appropriate filters for the incoming frames in the MAC_Packet_Filter register:
     a. Receive All
     b. Promiscuous mode
     c. Hash or Perfect Filter
     d. Unicast, multicast, broadcast, and control frames filter settings
  3. Program the following fields for proper flow control in the MAC_Q0_Tx_Flow_Ctrl register:
     a. Pause time and other Pause frame control bits
     b. Transmit Flow control bits
     c. Flow Control Busy
  4. Program the MAC_Interrupt_Enable register as required, and if applicable, for your configuration.
  5. Program the appropriate fields in the MAC_Configuration register for example, Inter-packet gap while transmission and jabber disabled.
  6. Set bits 0 and 1 in MAC_Configuration registers to start the MAC TX and RX.

- Performing Normal Receive and Transmit Operation

  For normal operation, complete the following steps:

  1. For normal TX and RX interrupts, read the interrupt status. Then, poll the descriptors, reading the status of the descriptor owned by the Host (either TX or RX).
  2. Set appropriate values for the descriptors, ensuring that TX and RX descriptors are owned by the DMA to resume the transmission and reception of data.
  3. If the descriptors are not owned by the DMA (or no descriptor is available), the DMA goes into the SUSPEND state. The transmission or reception can be resumed by freeing the descriptors and writing the descriptor tail pointer to TX/RX tail pointer register (DMA_CH[n]_TxDesc_Tail_Pointer and DMA_CH[n]_RxDesc_Tail_Pointer).
  4. The values of the current host TX or RX descriptor address pointer can be read for the debugging process (DMA_CH[n]_Current_App_TxDesc and DMA_CH[n]_Current_App_RxDesc).
  5. The values of the current host transmit buffer address pointer and receive buffer address pointer can be read for the debugging process (Register DMA_CH[n]_Current_App_TxBuffer and DMA_CH[n]_Current_App_RxBuffer).

- Stopping and Starting Transmission

  Complete the following steps to pause the transmission for some time:

1. Disable the TX DMA (if applicable) by clearing Bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) Register.
2. Wait for all the previous frame transmissions to complete. Check this by reading the appropriate bits of MTL_TxQ0_Debug Register (TRCSTS is not 01 and TXQSTS=0).
3. Disable the MAC TX and MAC RX by clearing Bit (RE) and Bit 1 (TE) of the MAC_Configuration Register.
4. Disable the RX DMA (if applicable) after making sure that the data in the RX FIFO is transferred to the system memory (by reading the appropriate bits of MTL_TxQ0_Debug Register, PRXQ=0 and RXQSTS=00).
5. Make sure that both TX Queue and RX Queue are empty (TXQSTS is 0 in MTL_TxQ0_Debug Register and RXQSTS is 0 in MTL_RxQ0_Debug Register).
6. To restart the operation, first start the DMAs, and then enable the MAC TX and RX.

- Programming Guidelines for Multi-Channel Multi-Queuing
  - Transmit

1. Program the TX queue size in the TQS field of MTL_TxQ[n]_Operation_Mode register. The size of the queue is determined based on the value programmed in TQS field. In the Transmit operation, the number of channels is equal to the number of the queues. For this reason, the Channel to Queue mapping is fixed.
2. For a queue to be used, the queue needs to be enabled in TXQEN in the corresponding MTL_TxQ[n]_Operation_Mode Register. In DMA configurations, ST bit of DMA_CH[n]_Tx_Control Register and the corresponding TXQEN in MTL_TxQ[n]_Operation Mode Register need to be enabled.
3. The scheduling method needs to be programmed in SCHALG of MTL_Operation_Mode register.
4. Program the MTL_TxQ[n]_Quantum_Weight for generic or DCB queue as per the selected algorithm. In case of CBS algorithm in AVB queues, the MTL_TxQ[n]_ETS_Control, MTL_TxQ[n]_SendSlopeCredit, MTL_TxQ[n]_HiCredit and MTL_TxQ[n]_LoCredit registers also need to be programmed as required.
5. If DCB is enabled and PFC function is required, program MAC_TxQ_Prty_Map0 Register to assign a fixed priority to the queue. This assigned priority is used to determine whether the corresponding queue should stop transmitting packet based on the received PFC packet.

  - Receive

1. Program the Receive queue size in the RQS field of MTL_RxQ[n]_Operation_Mode Register. Based on the value programmed in RQS field, the size of the queue is determined.
2. Enable the Receive Queues 0 to 7 in the fields RXQ0EN to RXQ7EN in MAC_RxQ_Ctrl0 Register for AV or DCB. In DMA configurations, SR bit of statically or dynamically mapped DMA_CH[n]_Rx_Control Register and corresponding RXQ[n]_EN in MAC_RxQ_Ctrl0 Register needs to be enabled.
3. The MAC routes the RX packets to the RX Queues based on following packet types:
   a. AV PTP Packets: Based on the programming of AVPTPQ in MAC_RxQ_Ctrl1 Register.
   b. AV Untagged Control packets: Based on the programming of AVCPQ in MAC_RxQ_Ctrl1 Register.
   c. Data Center Bridging (DCB) related Link Layer Discovery Protocol (LLDP) packets. Program DCBCPQ in MAC_RxQ_Ctrl1 Register to indicate to MAC which queue should get the DCB packets.
   d. VLAN Tag Priority field in VLAN Tagged packets: Program PSRQ7-0 of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 Register for the routing of tagged packets based on the USP (user Priority) field of the received packets to the RX Queues 0 to 7.
   e. The AV tagged control and data packets are also routed based on PSRQ field of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers.

4. If multiple RX DMA channels are enabled, the following programming should be done for proper arbitration and mapping:
   a. Program the RAA field of MTL_Operation_Mode register to select the arbitration algorithm to decide which RxQ is read out from the RxFIFO memory.
   b. Program the MTL_RxQ[n]_Control to decide the weights and the packet arbitration for each RxQ.
   c. If static mapping is programmed in MTL_RxQ_DMA_Map[n] register (RXQ[n]DADMACH is reset to 0), bits RXQx2DMA and others need to be programmed to select the channel for which each queue is mapped.
   d. Set RXQ[n]DADMACH bit in MTL_RxQ_DMA_Map0 Register to select dynamic mapping of packets in each RxQueue.
   e. In dynamic channel mapping, the routing of a packet to a specific RxDMA channel is decided by the value of DCS field in the lowest MAC Address Register.

- Programming Guidelines for IEEE 1588 Timestamping
  – Initialization Guideline for System Time Generation

    You can enable the timestamp feature by setting Bit 0 of the MAC_Timestamp_Control Register. However, it is essential that the timestamp counter should be initialized after this bit is set. Complete the following steps during DWC_ether_qos initialization:

    1. Mask the Timestamp Trigger interrupt by clearing Bit 16 of MAC_Interrupt_Enable Register.
    2. Set Bit 0 of MAC_Timestamp_Control Register to enable timestamping.
    3. Program MAC_Sub_Second_Increment Register based on the PTP clock frequency.
    4. If the Fine Correction approach is used, program MAC_Timestamp_Addend and set Bit 5 of MAC_Timestamp_Control Register.
    5. Poll the MAC_Timestamp_Control Register until Bit 5 is cleared.
    6. Program Bit 1 of MAC_Timestamp_Control Register to select the Fine Update method (if required).
    7. Program MAC_System_Time_Seconds_Update Register and MAC_System_Time_Nanoseconds_Update Register with the appropriate time values.
    8. Set Bit 2 in MAC_Timestamp_Control Register. The timestamp counter starts operation as soon as it is initialized with the value written in the Timestamp Update registers. If one-step timestamping is enabled:
       a. To enable one-step timestamping, program Bit 27 of the TDES3 Context Descriptor.
       b. Program registers MAC_Timestamp_Ingress_Asym_Corr and MAC_Timestamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
    9. Enable the MAC RX and TX for proper timestamping.

  – System Time Correction

    To synchronize or update the system time in one process (coarse correction method), complete the following steps:

    1. Set the offset (positive or negative) in the Timestamp Update registers (MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update).
    2. Set Bit 3 (TSUPDT) of the MAC_Timestamp_Control Register. The value in the Timestamp Update registers is added to or subtracted from the system time when the TSUPDT bit is cleared.

    To synchronize or update the system time to reduce system-time jitter (fine correction method), complete the following steps:

1. With the help of the algorithm explained in “System Time Register Module”, calculate the rate by which you want to make the system time increment slower or faster.
2. Update the MAC_Timestamp_Addend with the new value and set Bit 5 of the MAC_Timestamp_Control Register.
3. Wait for the time for which you want the new value of the Addend register to be active. You can do this by enabling the Timestamp Trigger interrupt after the system time reaches the target value.
4. Program the required target time in MAC_PPS[n]_Target_Time_Seconds Register and MAC_PPS[n]_Target_Time_Nanoseconds Register.
5. Enable the Timestamp interrupt in bit 12 of MAC_Interrupt_Enable register.
6. Set Bit 4 in Register MAC_Timestamp_Control.
7. When this trigger causes an interrupt, read MAC_Interrupt_Status Register.
8. Reprogram MAC_Timestamp_Addend Register with the old value and set Bit 5 again.

3.12.7 Peripheral Component Interconnect Express (PCIe) Controller

3.12.7.1 Overview

The PCIe controller adheres to the Intel® PIPE (PHY interface for the PCIe) interface, facilitating seamless integration with PIPE-compliant PHY. Moreover, the controller conforms to the AMBA® AXI4 specifications. It supports differential bus speeds of PCIe Gen1 (2.5 Gbps), PCIe Gen2 (5.0 Gbps) and PCIe Gen3 (8.0 Gbps).

3.12.7.2 Features

- Two ports: Port 0 and Port 1 (shared with USB Port 1)
- One port supports Root Complex (RC) mode and Endpoint (EP) mode; another port supports RC mode
- One port supports x2 link, another port supports x1 link
- Supports link rate of 2.5 GT/s, 5.0 GT/s, 8.0 GT/s, (only for x2 link port) per lane
- PCIe Base Specification Revision 3.0 compliant
- PIPE 4.0 compliant
- Supports memory, I/O, configuration, and message transactions.
- Supports maximum payload size: 256 bytes
- Supports maximum read request size: 512 bytes
- Supports 1 VC (Virtual Channel)
- AER (Advanced Error Reporting)
- ECRC (End to End Cycle Redundancy Check) generation and check support
- Lane reversal
- Polarity inverse
- Legacy PCI power management
- Native ASPM (Active State Power Management) L0s and L1 states
- L1PMSS (L1 Power Management Substates) with CLKREQ#
- LTR (Latency Tolerance Reporting)
- MSI (Message Signaled Interrupt) per function up to 32 and INT x (legacy interrupts)
- AHB and AXI interfaces:
  - 1 AHB slave interface for bridge configuration
  - 1 AXI master interface, supporting outstanding requests write (16)/read (24)

- 1 AXI slave interface, supporting outstanding requests write (4)/read (4)
- 128-bit data support for AXI master and slave interfaces

3.12.7.3 Block Diagram

The PCIe consists of 4 different layers: **PCIe layer, Bridge layer, AXI layer** and **Physical layer**, as depicted in the figure below.

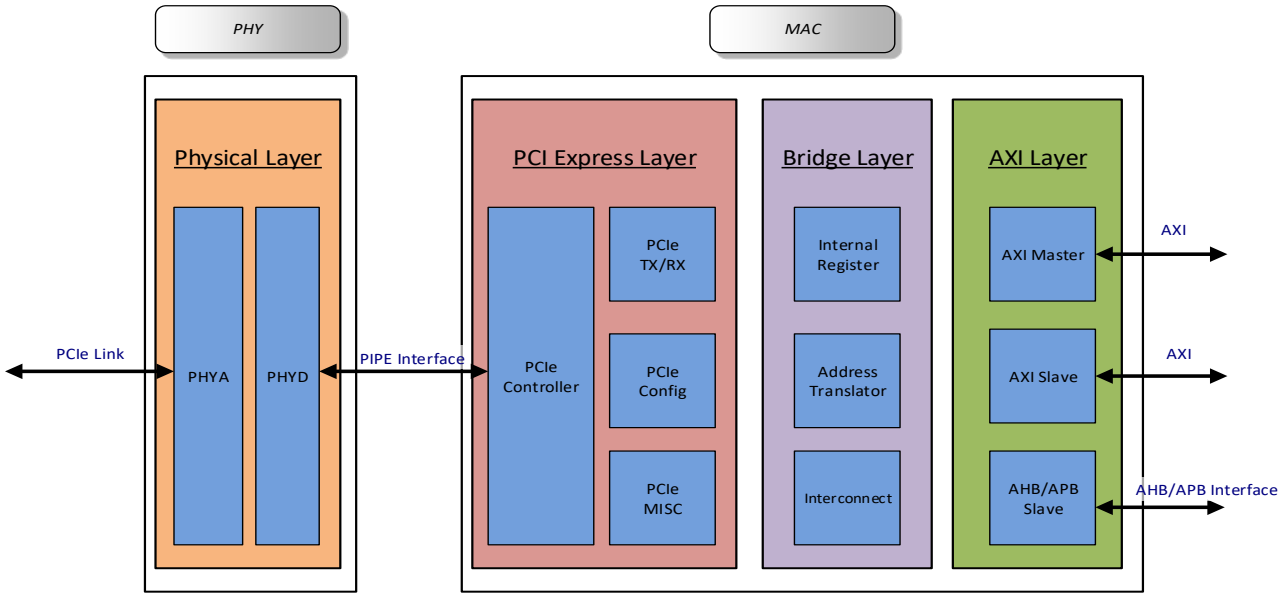


Figure 3-213 Block Diagram of PCIe Controller

Table 3-148 PCIe Layers and Components

Layer	Components
PCIe layer	• A PCIe controller, which is a PCIe core IP • A PCIe TX/RX interface between the bridge and the PCIe controller • A PCIe configuration interface to give the bridge layer access to the PCIe configuration space • A PCIe misc. interface to allow the bridge to manage low-power, interrupts, etc.
Bridge layer	• The internal registers of the PCIe controller • Address translator modules to convert between the AXI and PCIe interfaces • When transferring PCIe received requests to the AXI master, the address translator adds the corresponding AXI base address and forwards the request to the desired AXI master interface • The address translation method is similar when transferring AXI receive requests to the PCIe interface • An interconnect module to interconnect and arbitrate between input and output flow
AXI layer	• An AXI master interface, which manages requests and completions from the bridge layer to AXI • An AXI slave interface, which manages requests and completions from AXI to the bridge layer • An AHB or APB slave interface for bridge configuration
Physical layer	Please refer to Section 3.12.7.9

3.12.7.4 Function Description

3.12.7.4.1 I/O TLP Transfer

3.12.7.4.1.1 Transmitting I/O TLP

When operating in root port mode, the PCIe controller enables the local processor to send I/O request TLP via the AXI4 slave interface by configuring the address translation table appropriately.

- In cases where the target address of a write request from the AXI slave matches the address translation table for I/O TLP, the PCIe controller will transmit an I/O write TLP.
- Similarly, if the target address of a read request from the AXI slave matches the address translation table for I/O TLP, the PCIe controller will transmit an I/O read TLP.

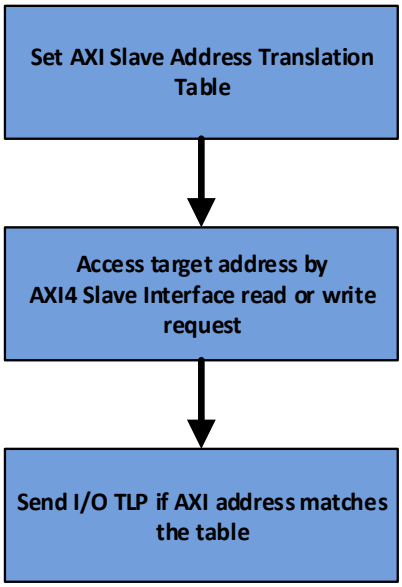


Figure 3-214 RC Send I/O TLP Flow

Table 3-149 RC Set Address Translation Table for I/O TLP Example

For AXI slave0 Table0: 0x0800 to 0x081F					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x800	ATR_IMPL	[0]	W	1'b1	The table is enabled.
	ATR_SIZE	[6:1]	W	6'd19	Table Size is 2^ (19+1) = 1MB
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] will be ignored to align the Table size
0x804	SRC_ADDR_MSB	[31:0]	W	32'hCCCCCCCC	Any AXI address in 0xCCCCCCCC_00000000 to 0xCCCCCCCC_000FFFFF range will match table0.
0x808	TRSL_ADDR_LSB	[31:12]	W	20'h05500	AXI address matching the table will be translated to IO Requested TLP with address 0x05500000 to 0x055FFFFF
0x80C	TRSL_ADDR_MSB	[31:0]	W	32'h0	The IO TLP does not need high 32-bit translated address.
0x810	TRSL_ID	[3:0]	W	4'd1	To the PCIe IO interface
	TRSF_PARAM	[18:16]	W	3'd2	IO TLP
	TRSF_PARAM	[19]	W	1'b0	NW flag is 0
	TRSF_PARAM	[22:20]	W	3'b0	No snoop
	TRSF_PARAM	[23]	W	1'b0	ECRC is not forwarded.
	TRSF_PARAM	[26:24]	W	3'b0	TC is 0.

3.12.7.4.1.2 Receiving I/O TLP

When PCIe controller operates in Endpoint mode and has declared I/O Type in its Base Address register (BAR) of configuration space, PCIe controller can receive I/O TLP from RC and forward it to the internal register or AXI Master interface depending on Address Translation Table Setting.

- When an IO Read TLP is received, it will be forwarded to AXI Master if destination of Address Translation Table for the I/O Window is the AXI master.
- When an IO Write TLP is received, it will be forwarded to Internal Register if destination of Address Translation Table for the I/O Window is the internal register.

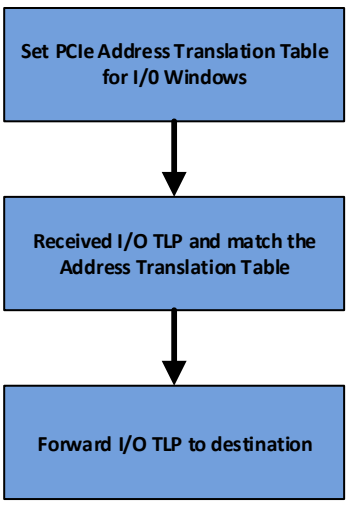


Figure 3-215 EP Send I/O TLP Flow

Table 3-150 EP Set Address Translation Table for I/O TLP Example

For PCIe Table0: 0x0600 to 0x061F					
Offset	**Register name**	**Bit Location**	**Action**	**Value**	**Description**
0x600	ATR_IMPL	[0]	W	1'b1	The table is enabled
	ATR_SIZE	[6:1]	W	6'd14	Table Size is 2^(14+1)=32KB
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] will be ignored to align the Table size
0x604	SRC_ADDR_MSB	[31:0]	W	32'b0	Any I/O address hit BAR2/3 and in 0x00000000_00000000 to 0x00000000_000007FF range will match table0
0x608	TRSL_ADDR_LSB	[31:12]	W	20'b0	TRSL_ADDR[ATR_SIZE:0] will be ignored to align the Table size
0x60C	TRSL_ADDR_MSB	[31:0]	W	32'h55555555	Any I/O address hit table0 will be mapped to 0x55555555_00000000 to 0x55555555_000007FFF
0x610	TRSL_ID	[3:0]	W	4'd5	To AXI Master 1
	TRSF_PARAM	[19:16]	W	4'd0	AXI Master ACACHE value is 0
	TRSF_PARAM	[20]	W	1'b0	AXI Master ALOCK value is 0
	TRSF_PARAM	[23:21]	W	3'b0	AXI Master APROT value is 0
	TRSF_PARAM	[27:24]	W	4'b0	AXI Master AQOS value is 0

3.12.7.4.2 Message TLP Transfer

The local processor is capable of transmitting message TLP via the AXI4 slave interface by configuring the address translation table accordingly. Once the PCIe controller receives a specific message TLP, it will notify the local processor via an *interrupt.Transmitting* message TLP.

3.12.7.4.2.1 Original Message Mechanism

In the default PLDA mode, the local processor sends a message via an AXI slave transaction, which is equivalent to a full message TLP. This includes the 4DW header that specifies the message code and routing information, as well as the payload if the message TLP contains data.

For instance, if the AXI slave transaction contains 12DW data, the first 4DW is the message header, while the last 8DW is reserved for message data. The local processor must ensure that the contents are properly prepared, including header format, message code, and message payload, within the 12DW data from the AXI slave.

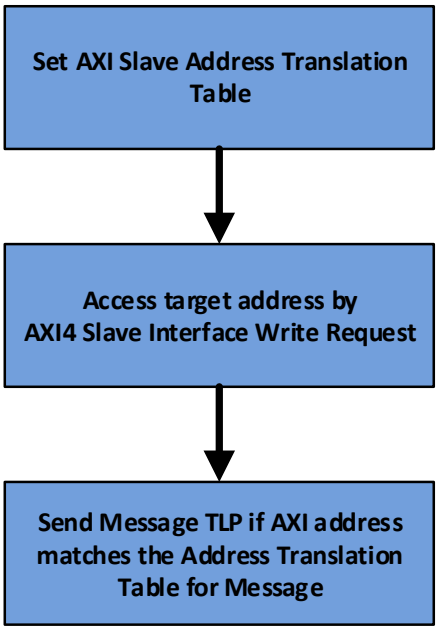


Figure 3-216 Transmit Message TLP Flow

Table 3-151 ADT Setting Example for Transmit Message TLP

For AXI slave0 Table0: 0x0800 to 0x081F					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x800	ATR_IMPL	[0]	W	1'b1	The table is enabled
	ATR_SIZE	[6:1]	W	6'd11	Table Size is 2^ (11+1) =4KB
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] will be ignored to align the table size
0x804	SRC_ADDR_MSB	[31:0]	W	32'hDDDDDDDD	Any AXI address in 0xDDDDDDDD_00000000 to 0xDDDDDDDD_00000FFF range will match table0.
0x808	TRSL_ADDR_LSB	[31:12]	W	20'h0	Message TLP does not need translated address
0x80C	TRSL_ADDR_MSB	[31:0]	W	32'h0	Message TLP does not need translated address
0x810	TRSL_ID	[3:0]	W	4'd0	To PCIe TX/RX interface
	TRSF_PARAM	[18:16]	W	3'd4	Message TLP
	TRSF_PARAM	[19]	W	1'b0	NW flag is 0
	TRSF_PARAM	[22:20]	W	3'b0	No snoop
	TRSF_PARAM	[23]	W	1'b0	ECRC is not forwarded
	TRSF_PARAM	[26:24]	W	3'b0	TC is 0

3.12.7.4.2.2 Message Pool Mechanism

In the message pool mode, the local processor sends a message, and the PCIe controller can aggregate several AXI slave transactions to form a complete Message TLP if those transactions match the Address Translation Table for Message. In this mode, the maximum length for a Message TLP is 8DW, including the 4DW header and a maximum of 4DW data.

For instance, suppose the local processor intends to send a message TLP with a length of 6DW, but a single AXI slave transaction contains a maximum of 2DW. In that case, the local processor can activate the message pool mode register and issue three AXI transactions, each with a length of 2DW, to transmit a 6DW message TLP.

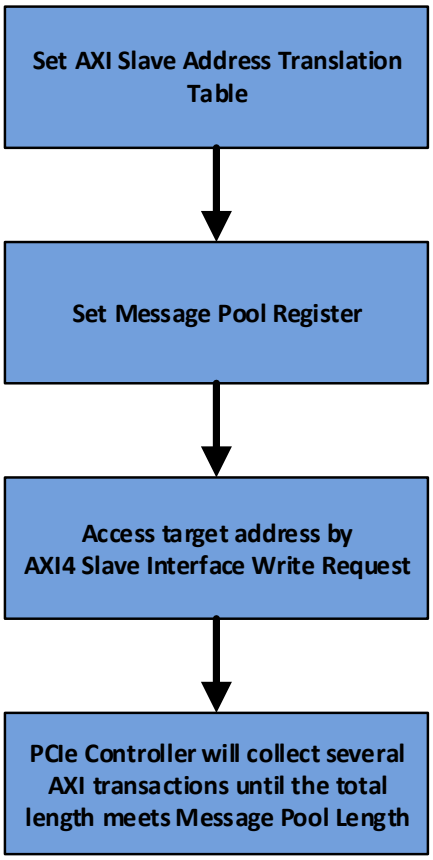


Figure 3-217 Transmit Message TLP by Message Pool Mode

Table 3-152 ADT Setting Example for Message Pool Mode

Offset	Register Name	Bit Location	Action	Value	Description
For AXI slave0 Table0: 0x0800 to 0x081F					
0x800	ATR_IMPL	[0]	W	1'b1	The table is enabled
	ATR_SIZE	[6:1]	W	6'd11	Table Size is 2^ (11+1) =4KB
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] will be ignored to align the Table size
0x804	SRC_ADDR_MSB	[31:0]	W	32'hDDDDDDDD	Any AXI address in 0xDDDDDDDD_00000000 to 0xDDDDDDDD_00000FFF range will match table0.
0x808	TRSL_ADDR_LSB	[31:12]	W	20'h0	Message TLP does not need translated address
0x80C	TRSL_ADDR_MSB	[31:0]	W	32'h0	Message TLP does not need translated address
0x810	TRSL_ID	[3:0]	W	4'd0	To PCIe TX/RX interface
	TRSF_PARAM	[18:16]	W	3'd4	Message TLP
	TRSF_PARAM	[19]	W	1'b0	NW flag is 0
	TRSF_PARAM	[22:20]	W	3'b0	No snoop
	TRSF_PARAM	[23]	W	1'b0	ECRC is not forwarded
	TRSF_PARAM	[26:24]	W	3'b0	TC is 0
For Message Pool Register: 0x01A8					
0x1A8	MSG_POOL_EN	[4]	W	1'b1	Enable Message Pool mode
	MSG_POOL_DLEN	[7:5]	W	3'd2	Message Data Payload length is 2DW and total length is 4DW+2DW = 6DW

3.12.7.4.2.3 Receiving Message TLP

The system can transmit incoming messages from the PCIe interface to either an internal register or an AXI master, depending on the message type and reception message register settings. To activate this message forwarding functionality, the software must properly configure the *"PMSG_RECEPTION_SETTINGS"* register, which ranges from 0x3F0 to 0x3FF.

The messages below are supported by the PCIe controller and forwarded to a dedicated destination.

- Unlock Message
- ATS Message
- LTR Message
- Optimized Buffer Flush/Fill
- Vendor Defined Type0
- Vendor Defined Type1

In the case of an LTR message, it is imperative that the message is forwarded to the internal register, and it must be directed to a fixed destination address of 0x1A4. Therefore, *LTR_DEST_ADDR* register of *"PMSG_RECEPTION_SETTINGS1 (0x3F4)"* must be fixed to 0x1A4.

For other supported messages, there is flexibility in whether they are forwarded to the internal register or the AXI master interface.

Field	Register	Description
PMSG_SUPPORT_RX	PMSG_RECEPTION_SETTINGS0 (0x3F0)	Enables the configuration of supported message types.
PMSG_DEST_ID	PMSG_RECEPTION_SETTINGS1 (0x3F4)	Enables selection of either the AXI master interface or internal register that will be used to forward messages.
PMSG_DEST_ADDR	PMSG_RECEPTION_SETTINGS2 (0x3F8) PMSG_RECEPTION_SETTINGS3 (0x3FC)	Allow for specifying the address when targeting an AXI master interface or internal register. • If "PMSG_DEST_ID" is set to the Internal Register (5'd12), then "PMSG_DEST_ADDR" must be fixed at 64'hCE0, and the PCIe controller can only support Message TLP with at most 8DW length. • If "PMSG_DEST_ID" is set to the AXI master interface (5'd4 to 5'd7), then the 12 LSBs of this address should be equal to 12'b0 and there are no limitations on the message length.

For other supported messages, there is the option to forward them either to the internal register or the AXI master interface.

Field	Register	Description
PMSG_SUPPORT_RX	PMSG_RECEPTION_SETTINGS0 (0x3F0)	Enables the configuration of supported message types.
PMSG_DEST_ID	PMSG_RECEPTION_SETTINGS1 (0x3F4)	Enables selection of either the AXI master interface or internal register to be utilized for message forwarding.

Field	Register	Description
PMSG_DEST_ADDR	PMSG_RECEPTION_SETTINGS2 (0x3F8) PMSG_RECEPTION_SETTINGS3 (0x3FC)	Allow for specifying the address when targeting an AXI master interface or internal register. • If "PMSG_DEST_ID" is set to Internal Register (5'd12), then "PMSG_DEST_ADDR" must be fixed at 64'hCE0. Additionally, the PCIe controller can only support Message TLP with at most 8DW length. • If "PMSG_DEST_ID" is set to the AXI master interface (5'd4 to 5'd7), then the 12 LSBs of this address should be set to 12'b0, and there are no limitations on the message length.

Once the PCIe controller receives an LTR message TLP, the message contents will be written into the internal register *"PCIE_LTR_VALUES" (0x1A4)*, and the status register *"LTR_MSG_RECEIVED" (0x14C [0])* will be set. This setting generates a Local Interrupt to notify the local processor.

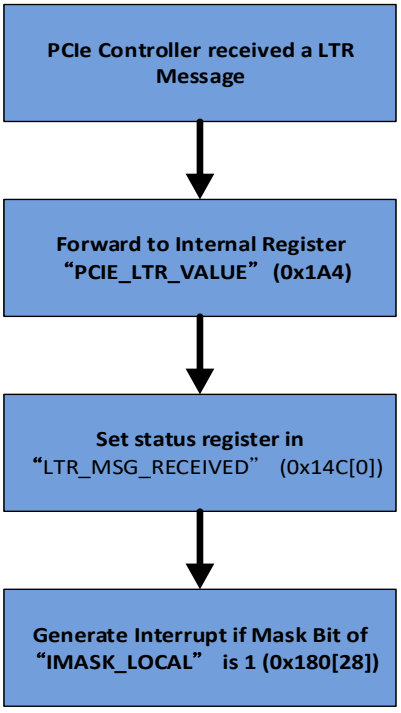


Figure 3-218 RC Received LTR Message Flow

Table 3-153 RC Handle LTR Message Interrupt Example

Clear RC LTR Message Interrupt					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x184	ISTATUS_LOCAL	[31:0]	R	32'h1000_0000	Check whether local interrupt status and Message TLP are received
0x14C	PCIE_MISC_STATUS	[31:0]	R	32'h0000_0001	Check what kind of Message is received
0x1A4	PCIE_LTR_VALIE	[31:0]	R	32'hxxxx_xxxx	Check LTR Value
0x14C	PCIE_MISC_STATUS	[31:0]	W	32'h0000_0001	After checking LTR value and performing requisite action, local processor can clear LTR status
0x184	ISTATUS_LOCAL	[31:0]	W	32'h1000_0000	Clear local status to de-assert local interrupt

When the PCIe controller receives a supported message TLP and the "PMSG_DEST_ID" is set to internal register, the entire message TLP contents, comprising both the header and data, will be written into the internal register *"PCIE_RECEIVED_MESSAGE" (0xCE0 to 0xCFF).* Additionally, the status *register "PCIE_MSG_RECEIVED" (0x14C [1])* will be set. This setting generates a local interrupt to notify the local processor.

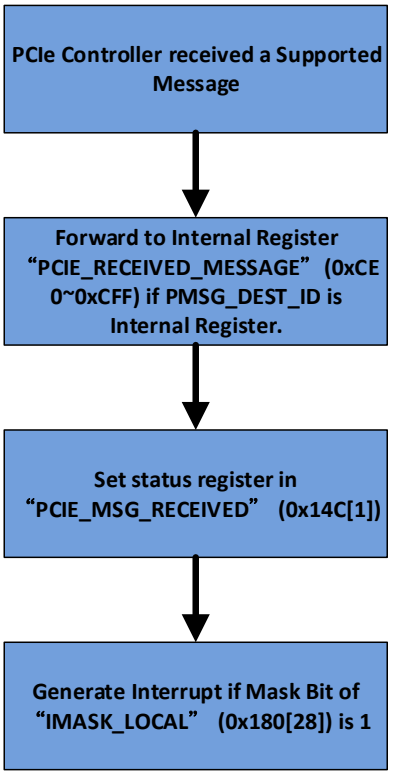


Figure 3-219 RC Received Other Supported Message Flow

Table 3-154 RC Handle Other Supported Message Interrupt Example

Clear RC LTR Message Interrupt					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x184	ISTATUS_LOCAL	[31:0]	R	32'h1000_0000	Check whether local interrupt status and message TLP are received
0x14C	PCIE_MISC_STATUS	[31:0]	R	32'h0000_0002	Check what kind of message is received
0xCE0 to 0xCFF	PCIE_RECEIVED_MESSAGE	[31:0]	R	32'hxxxx_xxxx	Check message header and data
0x14C	PCIE_MISC_STATUS	[31:0]	W	32'h0000_0002	After checking message type and performing requisite action, local processor can clear message status
0x184	ISTATUS_LOCAL	[31:0]	W	32'h1000_0000	Clear local status to de-assert local interrupt

3.12.7.4.3 Reset

The PCIe controller features two reset input ports: the power on reset and the *PERST#*, which is a fundamental reset defined in the PCIe specification. Additionally, the controller provides a software reset controller register, *"PCIE_RST_CTRL" (0x148)*, which can be utilized to reset various modules within the PCIe controller.

There are several PCIe link state transitions referred to as "Pl Exit Events." When a "Pl Exit Event" occurs, the PCIe controller automatically resets certain logic to clear the register back to its default value. The following describes the state transitions associated with the "Pl Exit Event":

Exit	Description
Dl Up Exit	LTSSM exits the Disable state or the link is unexpectedly down; then, LTSSM returns to the Detect.Quiet State.
Hot Reset Exit	LTSSM exits the Hot Reset State and returns to the Detect.Quiet state.
L2 Exit	LTSSM exits the L2 state and returns to the Detect.Quiet state.
LTSSM Exit	LTSSM enters the Detect.Quiet state from any state except the upper state.

- In **endpoint mode**, if a "Pl Exit Event" occurs, the PCIe controller will reset the PHY, physical layer, data link layer, and transaction layer, including the configuration space.
- In **Rootport mode**, if a "Pl Exit Event" occurs, the PCIe controller will reset the PHY, physical layer, data link layer, and transaction layer, with the exception of the configuration space.

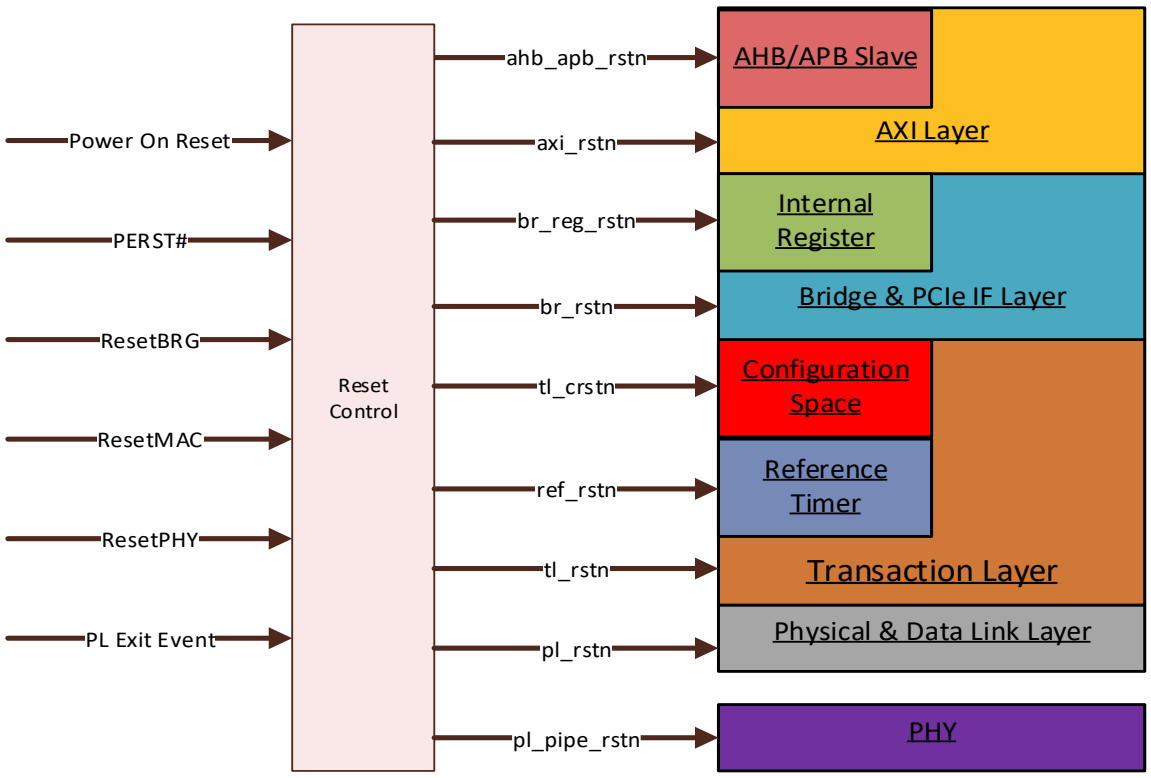


Figure 3-220 PCIe Reset Scheme

Table 3-155 Reset Signal Description

Reset Signal Name	Description
ahb_apb_rst_b	Clear AHB or APB slave logic in the AXI layer
axi_rstn	Clear AXI slave and AXI master logic in the AXI layer
br_reg_rstn	Clear internal register in the Bridge layer
br_rstn	Clear Bridge layer logic except the internal register
tl_crstn	Clears all configuration registers in the transaction layer, except sticky registers

Reset Signal Name	Description
tl_rstn	Clears all logic in the transaction layer except configuration registers and reference timer
ref_rstn	Clear reference timer
pl_rstn	Clears the physical and data-link layer logic
pl_pipe_rstn	Clear PHYD logic

Table 3-156 Reset Control Table

	Power on Reset	PERST# (Only for EP)	Pl Exit Event	ResetBRG (PCIE_RST_CTRL[2])	ResetMAC (PCIE_RST_CTRL[0])	ResetPHY (PCIE_RST_CTRL[1])
ahb_apb_rstn	Yes	Depend on PCIE_RST_CTRL[9]	Depend on PCIE_RST_CTRL[8]	Yes	No	No
axi_rstn	Yes	Depend on PCIE_RST_CTRL[9]	Depend on PCIE_RST_CTRL[8]	Yes	No	No
br_reg_rstn	Yes	Depend on PCIE_RST_CTRL[10]	No	No	No	No
br_rstn	Yes	Depend on PCIE_RST_CTRL[9]	Depend on PCIE_RST_CTRL[8]	Yes	No	No
tl_crstn	Yes	Depend on PCIE_RST_CTRL[11]	Depend on PCIE_RST_CTRL[6] (only for EP)	No	Yes	No
tl_rstn	Yes	Depend on PCIE_RST_CTRL[7]	Depend on PCIE_RST_CTRL[6]	No	Yes	No
ref_rstn	Yes	Depend on PCIE_RST_CTRL[7]	Depend on PCIE_RST_CTRL[6]	No	Yes	No
pl_rstn	Yes	Depend on PCIE_RST_CTRL[5]	Depend on PCIE_RST_CTRL[14]	No	Yes	No
pl_pipe_rstn	Yes	Depend on PCIE_RST_CTRL[4]	Depend on PCIE_RST_CTRL[12]	No	No	Yes

Table 3-157 Reset Scope for Pl Exit Event

Role	Dl Up Exit	Hot Reset Exit	L2 Exit
RC	Pl_rstn Tl_rstn	Pl_rstn Tl_rstn	Pl_rstn Tl_rstn

3.12.7.4.4 Handling Interrupts

The ISTATUS_LOCAL (located at *PCIE_MAC Base address + 0x0184*) register reports the interrupt source in bits 31 to 0 when any of the following events occurs. The host processor has the ability to enable or mask each interrupt source independently by setting or clearing the corresponding bit in the *IMASK_LOCAL* (located at *PCIE_MAC Base address + 0x0180*) register. PCIe has the capability to generate interrupts to the Local processor for the following events:

Bit	Description
Bit [31]	System error signaled
Bit [30]	PM/LTR/Hotplug event for RC

Bit	Description
Bit [29]	AER Event for RC
Bit [28]	Message TLP received except LTR and PTM
Bit [27]	Asserted when PCI interrupt line D is asserted
Bit [26]	Asserted when PCI interrupt line C is asserted
Bit [25]	Asserted when PCI interrupt line B is asserted
Bit [24]	Asserted when PCI interrupt line A is asserted
Bit [23]	L2 Wakeup: Asserted when L2 wakeup event happened
Bit [22]:	PCIe discard error: Asserted to signal a completion timeout on a PCIe read request
Bit [21]:	PCIe fetch error: Asserted to indicate that an error occurred on a PCIe read request
Bit [20]:	PCIe post error: Asserted to indicate that an error occurred on a PCIe write request
Bit [19]:	PCIe PTM message received: Asserted to indicate that a precise time message was received
Bit [18]:	AXI discard error: Asserted to signal a completion timeout on an AXI read request
Bit [17]:	AXI fetch error: Asserted to indicate that an error occurred on an AXI read request
Bit [16]:	AXI post error: Asserted to indicate that an error occurred on an AXI write request
Bit [15:8]:	Multiple MSI received: Report MSI events to the local processor. Bit number i corresponds to function number i
Bit [7:0]	Reserved for RC mode because DMA engines are not implemented

The occurrence of any of these events leads to the reporting of the interrupt source in the *"ISTATUS_LOCAL (0x184)"* register. The Host processor has the ability to enable or mask each interrupt source independently by setting or clearing the corresponding bit in the *"IMASK_LOCAL (0x180)"* register. For detailed information about these registers, refer to "*MT8395 Register Map*".

When an interrupt source is active and not masked, an interrupt is generated on the AXI domain and reported through the *"pcie_interrupt_out"* output port. The processing of this interrupt upon receipt by the Local processor is application-specific. However, in general, the local processor:

- Reads the *"ISTATUS_LOCAL (0x184)"* register to determine the source of the interrupt.
- Reads any other bridge configuration space status registers if required.
- Reads any PCIe configuration space registers if required.
- Reads any PCIe device status registers if required.
- Performs the requisite actions.
- Clears the *"ISTATUS_LOCAL (0x184)"* interrupt source by writing 1 to the corresponding bit.

Table 3-158 Local Interrupt Table for RC Mode

Bit	Interrupt source	Level2 Mark	Level2 status	Level1 Mark	Level1 Status
0	Internal DMA0 End	N/A	Internal DMA0 Status (0x420)	0x180[0]	0x184[0]
1	Internal DMA1 End	N/A	Internal DMA1 Status (0x460)	0x180[1]	0x184[1]
2	Internal DMA2 End	N/A	Internal DMA2 Status (0x4A0)	0x180[2]	0x184[2]
3	Internal DMA3 End	N/A	Internal DMA3 Status (0x4E0)	0x180[3]	0x184[3]
4	Internal DMA4 End	N/A	Internal DMA4 Status (0x520)	0x180[4]	0x184[4]
5	Internal DMA5 End	N/A	Internal DMA5 Status (0x560)	0x180[5]	0x184[5]
6	Internal DMA6 End	N/A	Internal DMA6 Status (0x5A0)	0x180[6]	0x184[6]
7	Internal DMA7 End	N/A	Internal DMA7 Status (0x5E0)	0x180[7]	0x184[7]

Bit	Interrupt source	Level2 Mark	Level2 status	Level1 Mark	Level1 Status
8	Internal DMA0 Error	N/A	Internal DMA0 Status (0x420)	0x180[8]	0x184[8]
	Receive MSI for MSI Funciton0	MSI Function0 Mark (0xC08)	MSI Function0 Status (0xC04)		
9	Internal DMA1 Error	N/A	Internal DMA1 Status (0x460)	0x180[9]	0x184[9]
	Receive MSI for MSI Funciton1	MSI Function1 Mark (0xC18)	MSI Function1 Status (0xC14)		
10	Internal DMA2 Error	N/A	Internal DMA2 Status (0x4A0)	0x180[10]	0x18410]
	Receive MSI for MSI Funciton2	MSI Function2 Mark ((0xC28)	MSI Function2 Status (0xC24)		
11	Internal DMA3 Error	N/A	Internal DMA3 Status (0x4E0)	0x180[11]	0x184[11]
	Receive MSI for MSI Funciton3	MSI Function3 Mark (0xC38)	MSI Function3 Status (0xC34)		
12	Internal DMA4 Error	N/A	Internal DMA4 Status (0x520)	0x180[12]	0x184[12]
	Receive MSI for MSI Funciton4	MSI Function4 Mark (0xC48)	MSI Function4 Status (0xC44)		
13	Internal DMA5 Error	N/A	Internal DMA5 Status (0x560)	0x180[13]	0x184[13]
	Receive MSI for MSI Funciton5	MSI Function5 Mark (0xC58)	MSI Function5 Status (0xC54)		
14	Internal DMA6 Error	N/A	Internal DMA6 Status (0x5A0)	0x180[14]	0x184[14]
	Receive MSI for MSI Funciton6	MSI Function6 Mark (0xC68)	MSI Function6 Status (0xC64)		
15	Internal DMA7 Error	N/A	Internal DMA7 Status (0x5E0)	0x180[15]	0x184[15]
	Receive MSI for MSI Funciton7	MSI Function7 Mark (0xC78)	MSI Function7 Status (0xC74)		
16	AXI ADT Event: Post Error	ADT to Local Mark (0x1AC[24])	AXI ADT Status (0x1E0~0x1EF)	0x180[16]	0x184[16]
	N/A	N/A	N/A		
17	AXI ADT Event: Fetch Error	ADT to Local Mark (0x1AC[25])	AXI ADT Status (0x1E0~0x1EF)	0x180[17]	0x184[17]
	N/A	N/A	N/A		
18	AXI ADT Event: Discard Error	ADT to Local Mark (0x1AC[26])	AXI ADT Status (0x1E0~0x1EF)	0x180[18]	0x184[18]
	N/A	N/A	N/A		
19	AXI ADT Event: Doorbell	ADT to Local Mark (0x1AC[27])	AXI ADT Status (0x1E0~0x1EF)	0x180[19]	0x184[19]
	Precise Time Message Received	N/A	PTM Register (0xD80~0xD93)		
20	PCIe ADT Event: Post Error	ADT to Local Mark (0x1AC[28])	PCIe ADT Status (0x1D8~0x1DF)	0x180[20]	0x184[20]
	N/A	N/A	N/A		
21	PCIe ADT Event: Fetch Error	ADT to Local Mark (0x1AC[29])	PCIe ADT Status (0x1D8~0x1DF)	0x180[21]	0x184[21]

Bit	Interrupt source	Level2 Mark	Level2 status	Level1 Mark	Level1 Status
	N/A	N/A	N/A		
22	PCIe ADT Event: Discard Error	ADT to Local Mark (0x1AC[30])	PCIe ADT Status (0x1D8~0x1DF)	0x180[22]	0x184[22]
	N/A	N/A	N/A		
23	PCIe ADT Event: Doorbell	ADT to Local Mark (0x1AC[31])	PCIe ADT Status (0x1D8~0x1DF)	0x180[23]	0x184[23]
	L2 Remote Wakeup	N/A	N/A		
24	Interrupt line A is asserted	N/A	N/A	0x180[24]	0x184[24]
25	Interrupt line B is asserted	N/A	N/A	0x180[25]	0x184[25]
26	Interrupt line C is asserted	N/A	N/A	0x180[26]	0x184[26]
27	Interrupt line D is asserted	N/A	N/A	0x180[27]	0x184[27]
28	Message Received	N/A	Message Status Register (0x14C)	0x180[28]	0x184[28]
	LTR Message Received	N/A	Message Status Register (0x14C)		
29	AER Event	N/A	N/A	0x180[29]	0x184[29]
30	Receive PME Message	N/A	N/A	0x180[30]	0x184[30]
31	Reset Event	N/A	N/A	0x180[31]	0x184[31]
	System Error	N/A	N/A		

3.12.7.4.4.1 Receiving INTX Message

In the root port mode, the PCIe controller is capable of generating a local interrupt to inform the local processor when it receives an *INTX* message from an endpoint. Upon receiving an *ASSERT INTX* message, the PCIe controller sets the *"INT_EVT"* bit in the *"ISTATUS_LOCAL (0x184)"* register, and this status remains asserted until the Root Complex (RC) receives a DEASSERT INTX message from the endpoint.

- If *"ISTATUS_CTRL[3] (0x1AC)"* equals 0, the local processor is responsible for clearing the corresponding *"INT_EVT"* status bit to complete the *INTX* handle flow.
- If *"ISTATUS_CTRL[3] (0x1AC)"* equals 1, the hardware automatically clears the corresponding *"INT_EVT"* status bit upon receiving a *DEASSERT INTX* message from the endpoint.

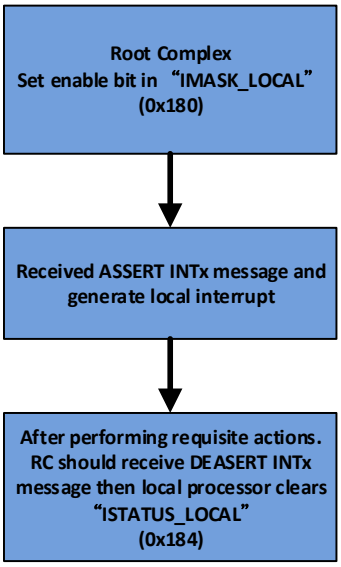


Figure 3-221 RC Handles Received INTX Flow If ISTATUS_CTRL[3] Is 0

3.12.7.4.4.2 Receiving MSI Message

In Root port mode, the PCIe Controller has the capability to support up to eight MSI capture addresses, and the local processor can set them to the *"IMSI_LO_ADDR (0xC00 + 0x10N, N = 0 to 7)"* and *"IMSI_HI_ADDR (0xC80 + 0x04N, N = 0 to 7)*" registers.

Whenever the core receives a memory write request at any of these addresses, it signals in the *"ISTATUS_LOCAL (0x184)"* register that an MSI has been received, and logs the received message number in the *"ISTATUS_MSI (0xC04 + 0x10*N, N = 0 to 7)"* register.

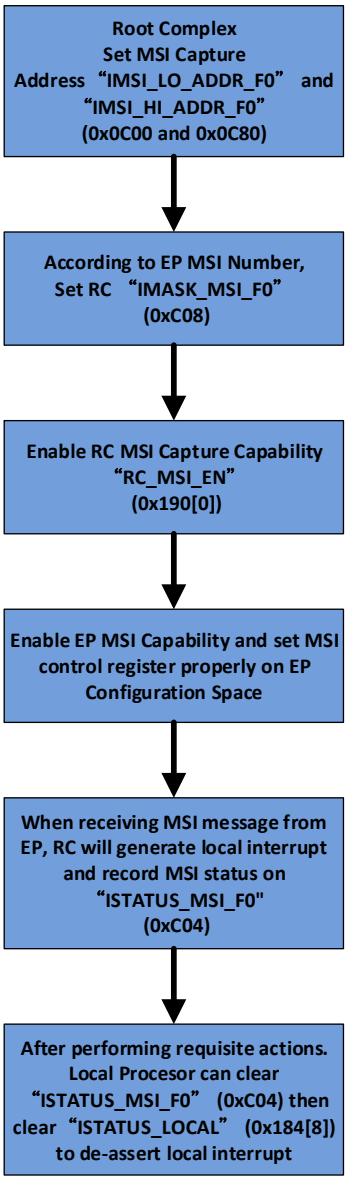


Figure 3-222 RC Handles Received MSI Flow

Table 3-159 RC Set MSI Capability Example

Set RC Function 0 MSI					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0xC00	IMSI_LO_ADDR_F0	[31:14]	W	32'hAAAABBB0	MSI address must be DW alignment; software must set bit[1:0] to 0.
0xC80	IMSI_HI_ADDR_F0	[31:0]	W	32'hFFFFFFFF	MSI capture address is 64'hFFFFFFFF_AAAABBB0. When MAC receives memory write TLP and its address is 64'hFFFFFFFF_AAAABBB0, MAC will write MSI data to 0xC04 of internal register.
0xC08	IMASK_MSI_ F0	[31:0]	W	32'h0000_000F	Support 4 MSI numbers.
0x190	RC_MSI_EN	[0]	W	1'b1	Enable RC Function0 MSI capture capability.
0x180	IMASK_LOCAL	[8]	W	1'b1	Set the local interrupt enable bit to generate a local interrupt when receiving a Function0 MSI.

Table 3-160 RC Clear MSI Status Example

Clear RC Function 0 MSI					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x184	ISTATUS_LOCAL	[31:0]	R	32'h0000_0100	Check whether local interrupt status and MSI message for function 0 are received.
0xC04	ISTAUS_MSI_F0	[31:0]	R	32'h0000_0004	Check MSI Status and received MSI number is 2.
0xC04	ISTAUS_MSI_F0	[31:0]	W	32'h0000_0004	After performing IRQ process, local processor can clear MSI status.
0x184	ISTATUS_LOCAL	[31:0]	W	32'h0000_0100	Clear local status to de-assert local interrupts

3.12.7.5 Power Management

The PCIe controller offers multiple low power modes, including L0s, L1, L1PM (L1 Power Management), L1SS (L1 Power Management Sub-State), and L2, to cater to various application and scenario requirements.

3.12.7.5.1 ASPM L0s Low Power

The ASPM L0s low-power state can be initiated by both the endpoint and rootport cores. This state does not require any application action, as it is automatically handled by the core.

- The core enters the ASPM L0s state after a pre-defined period of PCIe transmit inactivity, which is set by the ASPM L0s entry delay (in steps of 256ns from 1 - 31) defined by the *"PCIE_PEX_SPC2 (0x0D8)"* register.
- The core exits the ASPM L0s state as soon as a packet needs to be transmitted to the PCIe.

3.12.7.5.2 ASPM L1 Low Power

The core automatically handles the low-power state without requiring any action from the application.

- The core enters ASPM L1 after a specific period of inactivity, which is determined by the ASPM L1 entry delay. This delay is set by the *"PCIE_PEX_SPC2 (0x0D8)"* register and can be adjusted in increments of 256ns between 1 and 31.
- The core exits ASPM L1 when it needs to transmit a packet to the PCIe or when its link partner exits the low-power state.

Only endpoint cores can initiate the entry to the ASPM L1 low-power state.

- The endpoint core enters ASPM L1 after a specific period of inactivity, determined by the ASPM L1 entry delay set by the *"PCIE_PEX_SPC2 (0x0D8)"* register.
- The Rootport core enters ASPM L1 when requested by its link partner and there are no packets to be transmitted to the PCIe.
- Both cores exit ASPM L1 when a packet needs to be transmitted to the PCIe or when the link partner exits the low-power state.

Note:
If the downstream port (Rootport) declines the core's request to enter ASPM L1 after the specified period of activity, the link will remain in its full operational state. In this case, the endpoint waits for 10µs before requesting ASPM L1 entry again, following the guidelines of the PCIe Specification.

3.12.7.5.3 L1 Low Power

Only an endpoint core can initiate the entry to the L1 low-power state, but only when its legacy low-power state is not D0. The core automatically handles this low-power state and does not require any action from the application.

- To enter L1, an endpoint core must set its legacy low-power state to D1, D2, or D3, as indicated by the *"PCIE_ISTATUS_PM (0x19C)"* register. Any changes to the legacy power state are reported by the *"PM_EVT"* bit of the *"ISTATUS_LOCAL (0x184)"* register.
- The endpoint core enters L1 when there are no more packets to be transmitted. On the other hand, a Rootport core enters L1 when its link partner requests entry and there are no packets to be transmitted to the PCIe.
- Both cores exit L1 as soon as a packet needs to be transmitted.
- An endpoint core exits L1 either when directed to do so by its link partner or by setting the *"Send_PME"* bit in the *"PCIE_ICMD_PM (0x198)"* register. This sends a power management event to request that the link partner changes its legacy power state.

3.12.7.5.4 L2 Low Power

The L2 low-power state enables the PCIe link to be completely turned off, but it can only be initiated by a Rootport core.

- To enter the L2 state, a Rootport core disables the link as soon as its link partner is ready and the application sets the *"Turn_Off_Link"* bit in the *"PCIE_ICMD_PM (0x198)"* register.
- The Rootport core exits the L2 state and re-enables the link when the application clears the *"Turn_Off_Link"* bit in the "*PCIE_ICMD_PM (0x198)"* register.
- In contrast, an endpoint core automatically enters the L2 state when directed to do so by its link partner. It can only exit L2 if directed to do so by its link partner or by setting the *"WAKE_N"* bit in the *"PCIE_MISC_CTRL (0x348)"* register. The latter option sends a request to the link partner to re-enable the link.

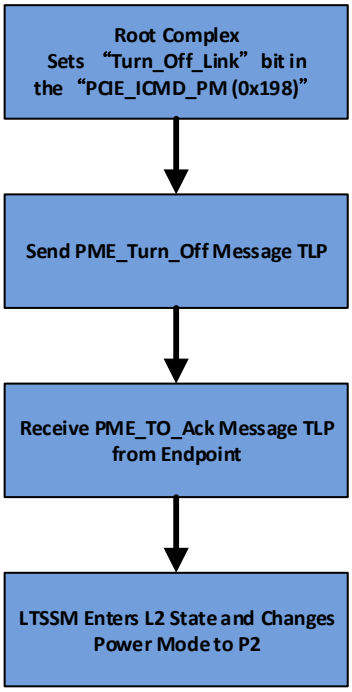


Figure 3-223 RC Enters L2 State Flow

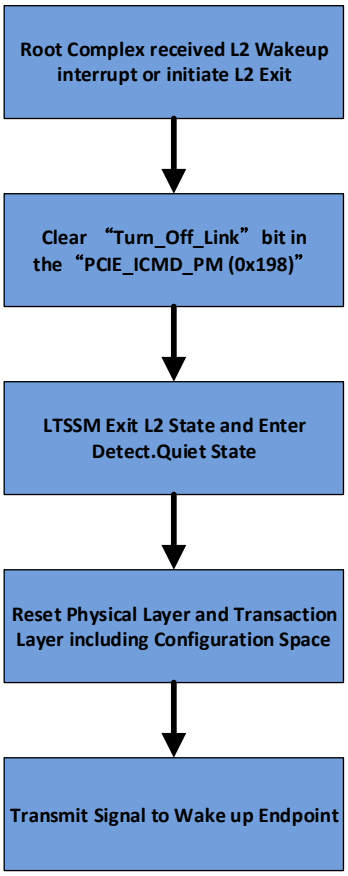


Figure 3-224 RC Exits L2 State Flow

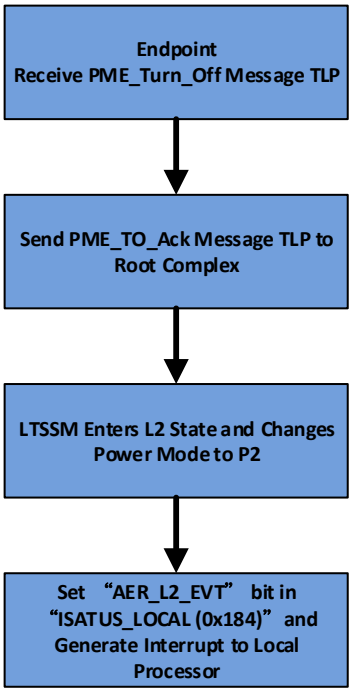


Figure 3-225 EP Enters L2 Flow

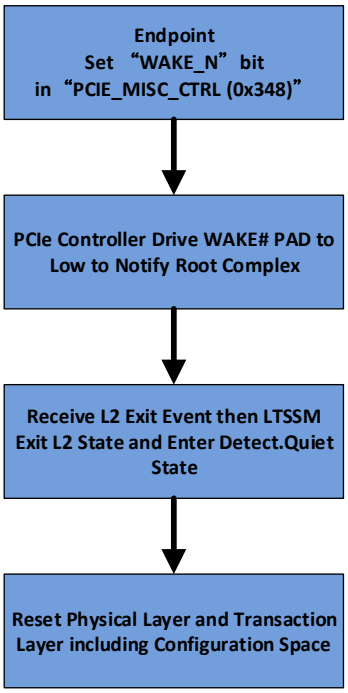


Figure 3-226 EP Exits L2 State Flow

3.12.7.5.5 Implementing Power Management with CLKREQ#

CLKREQ# is an optional side-band pin that may be available in certain form factors, and is utilized to reduce power consumption. This feature is supported by the clock power management and L1 PM substates with CLKREQ# features. It should be noted that only one of these power-saving techniques can be used at any given time.

The functionality of CLKREQ# is implemented through the link capability and link control registers present in the configuration space. When multiple functions are enabled, each function must be capable of utilizing CLKREQ#.

When CLKREQ# is employed to manage power consumption in the L2 state, the MAC transitions from the P0 to the P2 low power state to halt the reference clock, and subsequently deasserts CLKREQ#.

In the event that bit [8] of the *"PCIE_ICMD_PM (0x198)"* register for RC (RC CLKREQ# Clock control) is set to 1 upon entry into the L1 state, the MAC transitions directly from P0 to P2, bypassing P1. Upon exiting the L1 or L2 states, CLKREQ# is asserted to restart the Reference Clock.

The waveform below demonstrates an L1 entry where the application does not allow clock removal (bit [8] of the *"PCIE_ICMD_PM (0x198)" register = 0*), followed by an L1 entry where clock removal is permitted (bit [8] of the *"PCIE_ICMD_PM (0x198)" register = 1*).

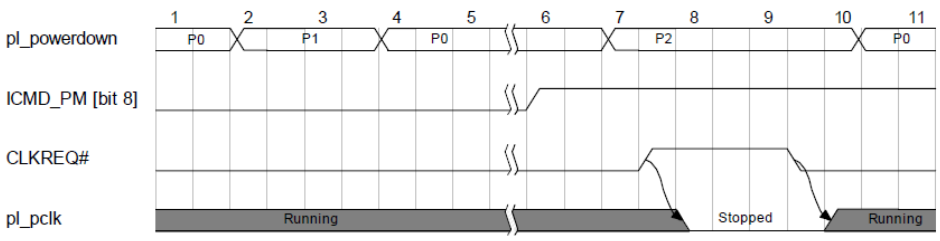


Figure 3-227 RC L1 Entry with and without Clock Removal

3.12.7.5.5.1 L1PM Sub-States with CLKREQ#

L1 PM sub-states (L1SS) is an extension to the PCIe Specification that permits further power savings in the L1 and ASPM L1 states. This feature is configured through the L1 PM sub-states capability and the *T_POWEROFF* parameter found in bits [7:5] of the *"PCIE_PEX_L1SS (0x0E0)"* register.

If any L1PM Substate enable bit in the configuration space is set to 1 while the core is in the L1 state, it has the ability to transition to the L1.1 or L1.2 sub-states if all the necessary conditions for this transition are met.

In the event that the L1PM Substate enable bit in the configuration space is set to 0 upon L1 entry, the core will remain in the L1.0 state. The core's present state is indicated by bits [10:8] of the *"PCIE_ISTATUS_PM (0x19C)"* register (l1pm_sm).

The PIPE clock *(PL_PCLK)* may be halted when the core is in the L1.1 or L1.2 sub-states. However, the Transaction Layer clock *(TL_CLK)* must continue to operate, albeit at a very low frequency, so that any traffic on the transaction layer transmit interface will prompt the core to assert CLKREQ# and exit low-power mode.

If the PCIe registers (PCIe configuration space registers or bridge registers) are accessed via the PCIe link while the core is in L1, L1.1, or L1.2, the core must exit the low power states. Conversely, if the registers are accessed via the AXI, the core does not need to exit low power mode.

Table 3-161 RC L1 State Setting

L1 Power State	RC CLKREQ# Clock Control 0x198[8]	L1PM Sub-State Enable 0x1118[3:0]	Root Complex Power State and CLKREQ# Status
L1 with P1	0x198[8]==0	0x1118[3:0]==0	LTSSM = L1 Power State = P1 L1PM State = L1.0 CLKREQ# Asserted (CLKREQ# is 0)
L1 with P2 (L1PM) (L1 with clock power management)	0x198[8]==1	0x1118[3:0]==0	LTSSM = L1 Power State = P2 L1PM State = L1.0 CLKREQ# De-asserted (CLKREQ# is 1)
L1SS	0x198[8]==0	0x1118[3:0]!=0	LTSSM = L1 Power State = P2 L1PM State = L1.1 or L1.2 CLKREQ# De-asserted (CLKREQ# is 1)
L1SS	0x198[8]==1	0x1118[3:0]!=0	LTSSM = L1 Power State = P2 L1PM State = L1.1 or L1.2 CLKREQ# De-asserted (CLKREQ# is 1)

3.12.7.6 Theory of Operations

3.12.7.6.1 Address Translation Mechanism

The PCIe controller adopts address translation to:

- Convert PCIe read and write requests to any AXI4 master interface read and write transaction.
- Convert any AXI4 slave interface read and write transaction to PCIe read and write requests.
- Convert an address between PCIe domain and AXI domain.

3.12.7.6.1.1 Address Translation for PCIe to AXI Direction

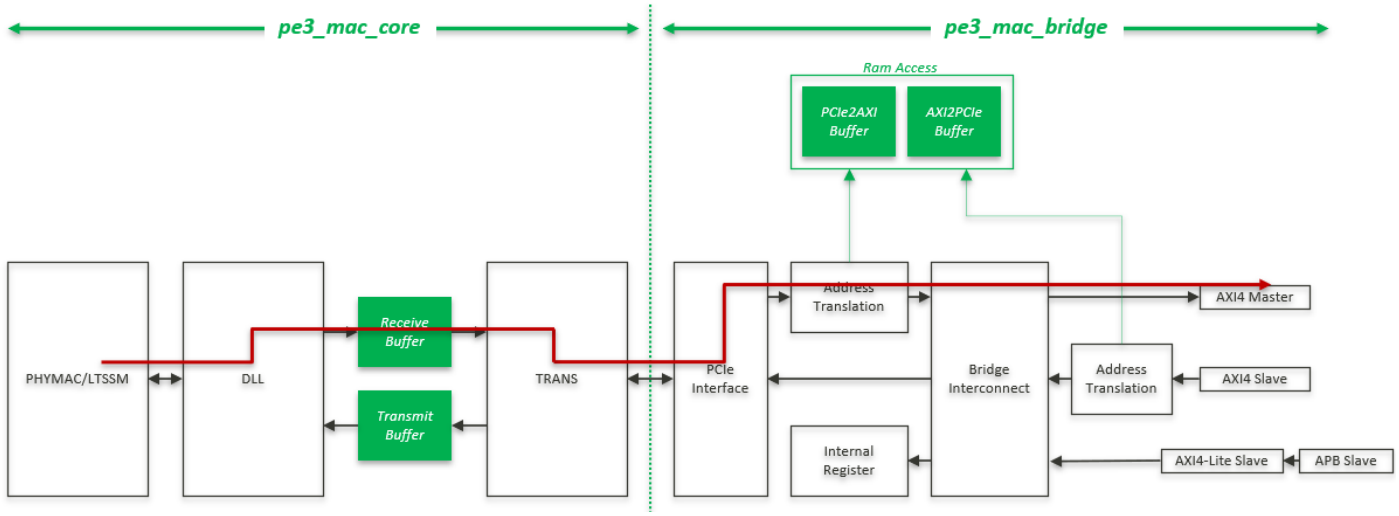


Figure 3-228 RC or EP Receive Memory Write TLP

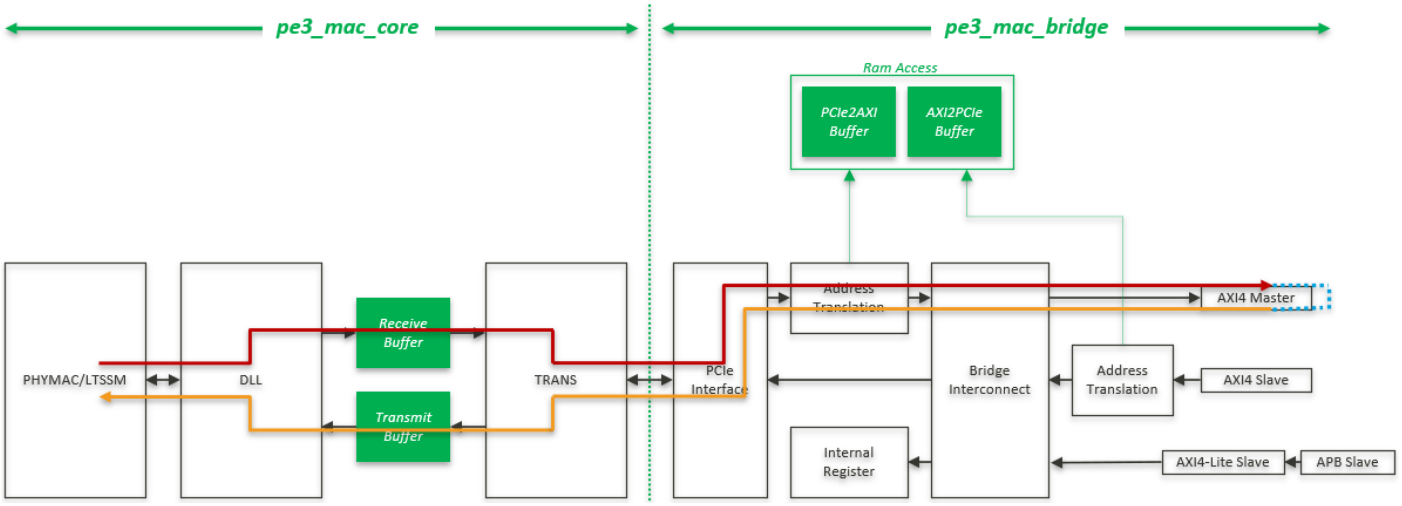


Figure 3-229 RC or EP Receive Memory Read TLP

3.12.7.6.1.1.1 RC mode

In the Root port mode, the core allows the implementation of up to sixteen translation tables.

During the transfer of PCIe received requests to the AXI master, the bridge performs a windows match utilizing the PCIe 64-bit address. Once a match is detected, the Bridge proceeds to forward the request to the intended AXI4 master interface, along with the corresponding AXI base address.

For instance, in the diagram below, if a write request is received at address 64'hBBBBBBBB_00020140, the bridge matches the address with the PCIe window's tables. Upon finding a match in Figure 3-230, the request is forwarded to the AXI4-Master #0 interface located at address 64'hCCCCCCCC_00000000 + 17'h0140.

The diagram below illustrates the PCIe to AXI4 master address translation in the root port mode.

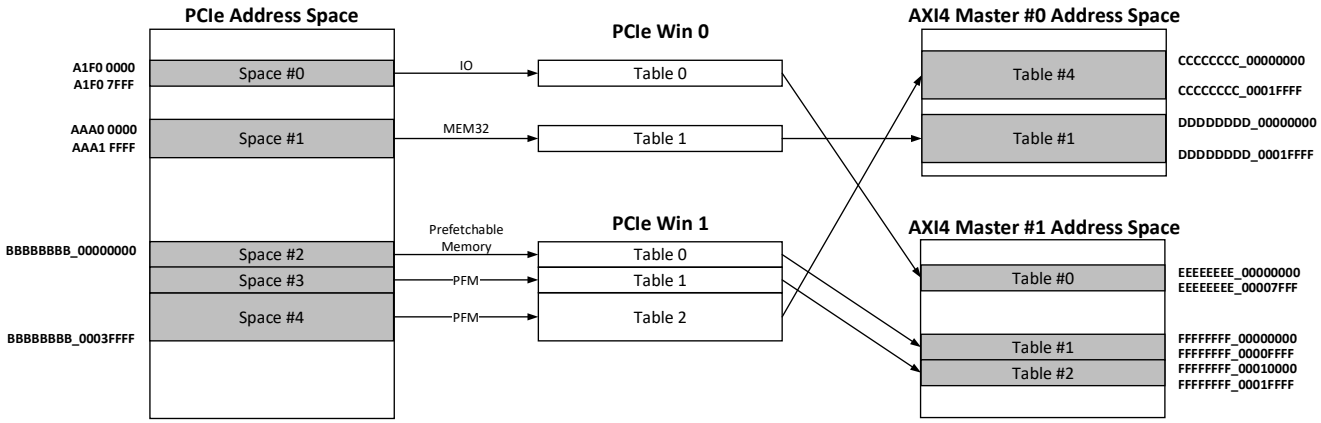


Figure 3-230 PCIe to AXI Master Address Translation for RC

3.12.7.6.1.1.2 EP Mode with Single Function

When the Core is in Endpoint mode with Single Function, you can implement up to two PCIe Windows.

Normal BAR0/1

- PCI Express BAR0/1 belong to the internal register and do not need to set any Translation Table.
  - BAR0/1 must be 64-bit mode and cannot be programmed to 32-bit mode.
- When PCIe Window #0 is implemented, PCI Express BAR2/3 belong to PCIe Window #0. PCIe read and write requests targeting BAR2/3 are then routed to PCIe Window #0 Address Translation module.
  - BAR2/3 can be programmed to 32-bit mode. It will be divided into BAR2 and BAR3.
- When PCIe Window #1 is implemented, PCI Express BAR4/5 belong to PCIe Window #1. PCIe read and write requests targeting BAR4/5 are then routed to PCIe Window #1 Address Translation module.
  - BAR4/5 can be programmed to 32-bit mode. It will be divided into BAR4 and BAR5.
- Each PCIe Window can support up to eight Translation Tables.

Swap BAR0/1 with PCIe Window #0

- When PCIe Window #0 is implemented, PCI Express BAR0/1 belong to PCIe Window #0. PCIe read and write requests targeting BAR0/1 are then routed to PCIe Window #0 Address Translation module.
  - BAR0/1 can be programmed to 32-bit mode. It will be divided into BAR0 and BAR1.
- PCI Express BAR2/3 belong to the internal register and do not need to set any Translation Table.
  - BAR2/3 must be 64-bit mode and cannot be programmed to 32-bit mode.
- When PCIe Window #1 is implemented, PCI Express BAR4/5 belong to PCIe Window #1. PCIe read and write requests targeting BAR4/5 are then routed to PCIe Window #1 Address Translation module.
  - BAR4/5 can be programmed to 32-bit mode. It will be divided into BAR4 and BAR5.
- Each PCIe Window can support up to eight Translation Tables.

Swap BAR0/1 with PCIe Window #1

- When PCIe Window #1 is implemented, PCI Express BAR0/1 belong to PCIe Window #1. PCIe read and write requests targeting BAR0/1 are then routed to PCIe Window #1 Address Translation module.
  - BAR0/1 can be programmed to 32-bit mode. It will be divided into BAR0 and BAR1.

- When PCIe Window #0 is implemented, PCI Express BAR2/3 belong to PCIe Window #0. PCIe read and write requests targeting BAR2/3 are then routed to PCIe Window #0 Address Translation module.
  - BAR2/3 can be programmed to 32-bit mode. It will be divided into BAR2 and BAR3.
- PCI Express BAR4/5 belong to internal register and do not need to set any Translation Table.
  - BAR4/5 must be 64-bit mode and cannot be programmed to 32-bit mode.
- Each PCIe Window can support up to eight Translation Tables.

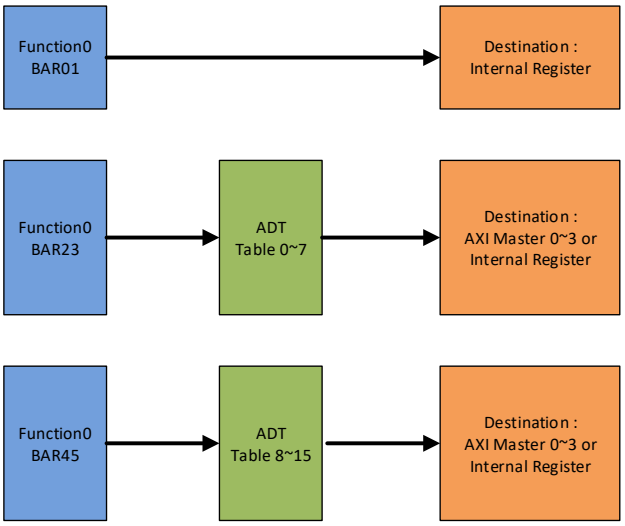


Figure 3-231 BAR and Address Translation Table Mapping in Single Function

When transferring PCI Express receive requests to the AXI master, the Bridge automatically removes the decoded BAR base address, and then performs window matching using the PCIe Window offset address. If a match is found, the Bridge then forwards the request to the desired AXI4 Master Interface and adds the corresponding AXI base address.

For example, in the diagram below, if a write request is received at address 64'hBBBBBBBB_00020140, the Bridge removes the upper 46-bit address bits, and then compares the remaining address 18'h20140 with the PCIe Window #1's tables. The request matches Figure 3-232, and is then forwarded to the AXI4-Master #1 interface, at address 64'hFFFFFFFF_00000000 + 16'h0140.

The following diagram shows PCIe to AXI4 Master Address Translation in Endpoint mode.

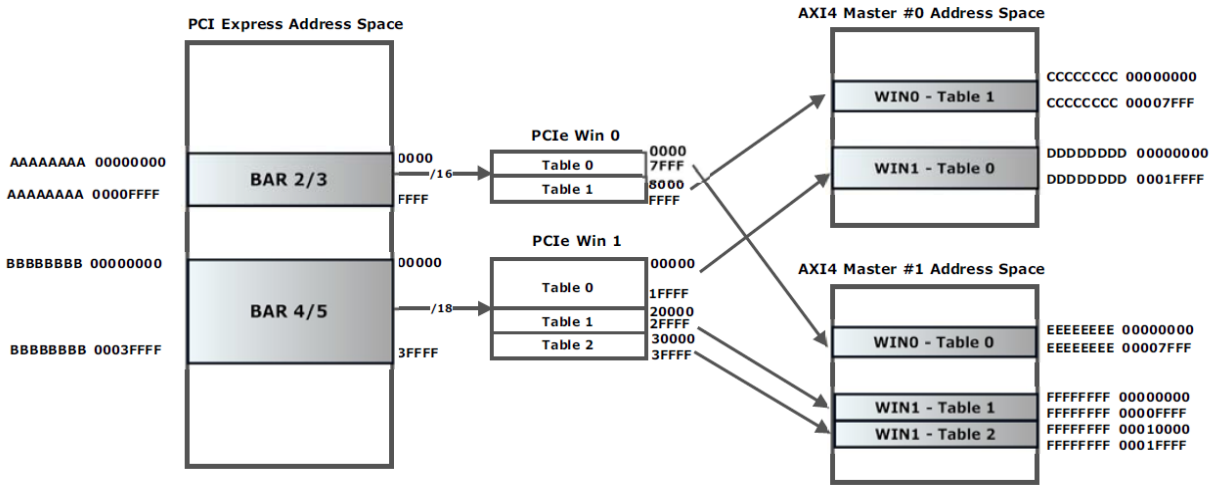


Figure 3-232 PCIe to AXI Master Address Translation for EP

3.12.7.6.1.2 Address Translation for AXI to PCIe Direction

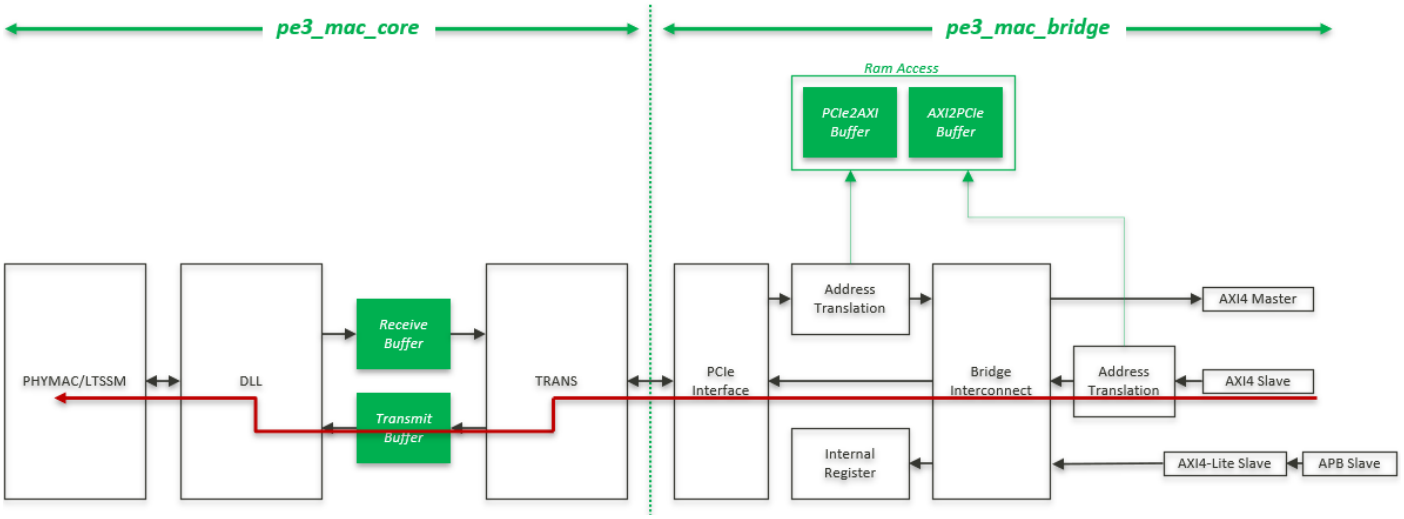


Figure 3-233 RC Initiated Memory Write

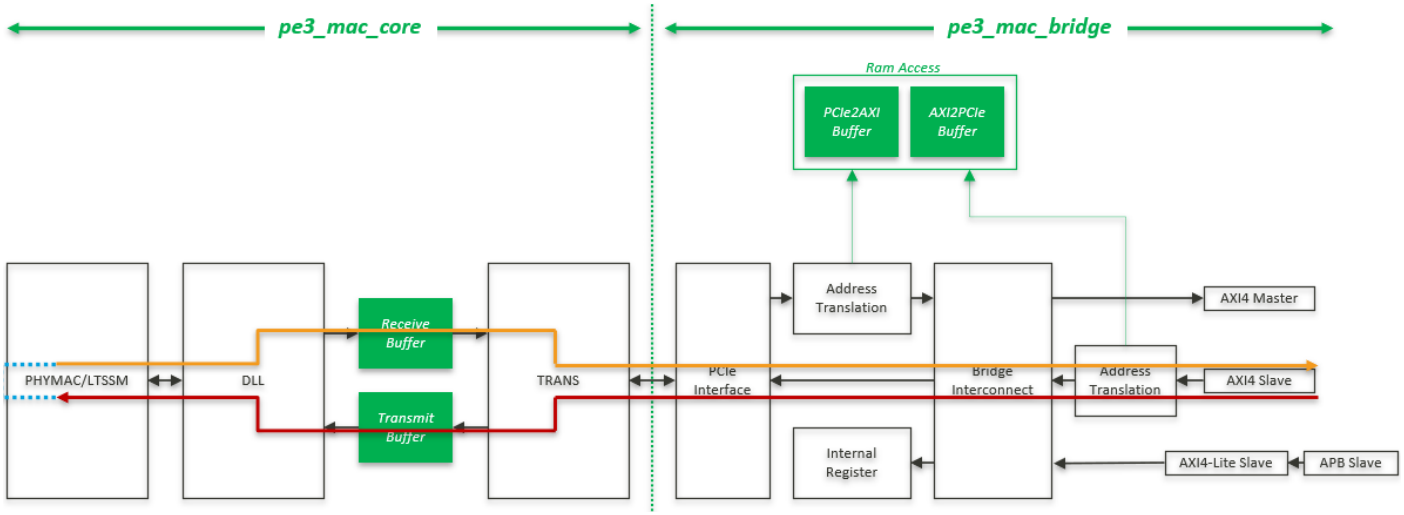


Figure 3-234 RC Initiated Memory Read

The address translation method utilized for transferring AXI receive requests to the PCIe interface is comparable. The core allows for up to eight translation tables to be implemented per implemented AXI4-slave interface.

The following diagram illustrates the AXI4 slave to PCIe address translation.

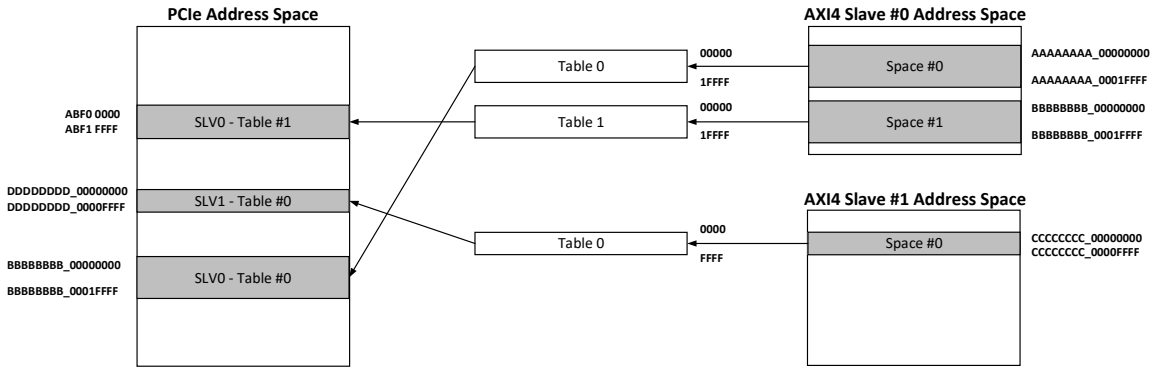


Figure 3-235 AXI Slave to PCIe Address Translation for RC

3.12.7.6.1.2.1 Set Address Translation Table

3.12.7.6.1.2.1.1 Address Translation Register

The address translation registers are categorized into six sections based on the internal bus slave port to be translated, as presented in Table 3-162.

Address translation settings allow for specification of the following parameters:

- Implementation or disablement of the table.
- Table size, ranging from 4 Kbytes to 16 Exabytes (2^64 bytes). If the table size is set to 16 Exabytes, no address conversion is performed since window matching is always successful. This setting is useful when address translation is required outside of the bridge.
- Source address, defining the offset address of the table inside the PCIe or AXI slave Window.
- Translated interface, specifying the interface to which the read or write request should be transferred. It can be set to PCIe, AXI4-Master, or AXI4-Stream.
- Transfer parameters, enabling definition of specific PCIe or AXI attributes or parameters per table.

Table 3-162 Address Translation Table Address

Byte Address	Read/Write (R/W)	Description
0x0600 – 0x06FF	Read only (RO) or RW	ATR_PCIE_WIN0: PCIe window 0 address translation tables 0-7
0x0700 – 0x07FF	RO or RW	ATR_PCIE_WIN1: PCIe window 1 address translation tables 0-7
0x0800 – 0x08FF	RO or RW	ATR_AXI4_SLV0: AXI4 slave 0 address translation tables 0-7
0x0900 – 0x09FF	RO or RW	ATR_AXI4_SLV1: AXI4 slave 1 address translation tables 0-7
0x0A00 – 0x0AFF	RO or RW	ATR_AXI4_SLV2: AXI4 slave 2 address translation tables 0-7
0x0B00 – 0x0BFF	RO or RW	ATR_AXI4_SLV3: AXI4 slave 3 address translation tables 0-7

Table 3-163 Transfer Parameter

Targeted Interface	Target ID	Description
PCIe	4'h0	• Bit [2:0]: TLP type – 3'b000: Memory – 3'b001: Memory locked – 3'b011: Translation request – 3'b101: Memory translated request – 3'b010: I/O – 3'b100: Message – Other values are reserved. • Bit [3]: Translation request no write (NW) flag (must be 0b when the TLP type is different from 3'b011). • Bit [6:4]: TLP attributes – Bit 4: No snoop – Bit 5: Relaxed ordering – Bit 6: ID-based ordering • Bit [7]: ECRC forward • Bit [10:8]: Traffic class

Targeted Interface	Target ID	Description
		• Bit [11]: Reserved
AXI4-Master	4'h4 4'h7	• Bit [3:0]: ACACHE • Bit [4]: ALOCK • Bit [7:5]: APROT • Bit [11:8] AQOS
AXI4-Stream	4'h8 4'hB	• Bit [7:0]: TID • Bit [11:8]: TDEST

3.12.7.6.1.2.2 Programming Address Translation Table (PCIe to AXI Master Direction)

Table 3-164 Set Address Translation Table for PCIe to AXI Direction

For Table0: 0x0600 to 0x061F					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x600	ATR_IMPL	[0]	W	1'b1	The table is enabled.
	ATR_SIZE	[6:1]	W	6'd14	Table Size is 2^ (14+1) =32KB.
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] will be ignored to align the Table size.
0x604	SRC_ADDR_MSB	[31:0]	W	32'b0	Any memory address in 0x00000000_00000000 to 0x00000000_000007FFF range will match table0.
0x608	SRC_BAR	[3:0]	W	4'd0	If the device has multiple functions, it is necessary to set dedicated BAR number for the table. If the device has only a single function, the register would be ignored by the bridge.
0x608	SRC_FUNC	[8:4]	W	5'd0	If the device has multiple functions, it is necessary to set dedicated function number for the table. If the device has only one single function, the register would be ignored by the bridge.
0x608	TRSL_ADDR_LSB	[31:12]	W	20'b0	TRSL_ADDR[ATR_SIZE:0] will be ignored to align the table size
0x60C	TRSL_ADDR_MSB	[31:0]	W	32'hEEEEEEEE	Any memory address hit table0 is mapped to 0xEEEEEEEE_00000000 to 0xEEEEEEEE_000007FFF.
0x610	TRSL_ID	[3:0]	W	4'd4	To AXI master 0
	TRSF_PARAM	[19:16]	W	4'd0	AXI master ACACHE value is 0.
	TRSF_PARAM	[20]	W	1'b0	AXI master ALOCK value is 0.
	TRSF_PARAM	[23:21]	W	3'b0	AXI master APROT value is 0.
	TRSF_PARAM	[27:24]	W	4'b0	AXI master AQOS value is 0.

3.12.7.6.1.2.3 Programming Address Translation Table (AXI slave to PCIe Direction)

AXI slave0 uses 2 tables to map to 2 destinations.

- Space0 start address: 0xAAAAAAAA_00000000
- As the table0 size of AXI slave0 is 17 bits,
  if AXI slave0 address range is between 0xAAAAAAAA_00000000 and 0xAAAAAAAA_0001FFFF, it will match table 0 and be translated to new memory addresses 0xBBBBBBBB_00000000 to 0xBBBBBBBB_0001FFFF.

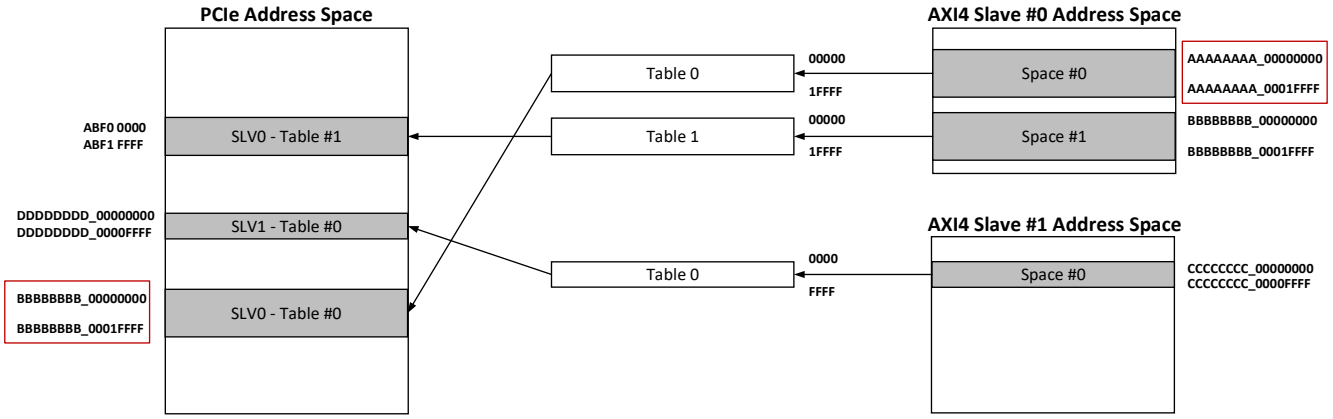


Figure 3-236 AXI Slave to PCIe Address Translation

Table 3-165 Set Address Translation Table for AXI to PCIe Direction

For AIX Slave0 Table0: 0x0800 to 0x081F					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x800	ATR_IMPL	[0]	W	1'b1	The table is enabled.
	ATR_SIZE	[6:1]	W	6'd16	Table Size is 2^ (16+1) =128KB.
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] is ignored to align the table size.
0x804	SRC_ADDR_MSB	[31:0]	W	32'hAAAAAAAA	Any AXI address between 0xAAAAAAAA_00000000 to 0xAAAAAAAA_00001FFFF will match table0.
0x808	TRSL_ADDR_LSB	[31:12]	W	20'b0	TRSL_ADDR[ATR_SIZE:0] will be ignored to align the Table size.
0x80C	TRSL_ADDR_MSB	[31:0]	W	32'hBBBBBBBB	Any AXI address hits table0 will be mapped to 0xBBBBBBBB_00000000 to 0xBBBBBBBB_000001FFFF.
0x810	TRSL_ID	[3:0]	W	4'd0	To PCIe interface
	TRSF_PARAM	[18:16]	W	3'd0	Memory TLP
	TRSF_PARAM	[19]	W	1'b0	NW flag is 0
	TRSF_PARAM	[22:20]	W	3'b0	No snoop
	TRSF_PARAM	[23]	W	1'b0	ECRC is not forwarded.
	TRSF_PARAM	[26:24]	W	3'b0	TC is 0.

3.12.7.6.2 Access Configuration Space and Configure TLP Transfer

The CPU located in a specific locality can access its own configuration space, as well as the configuration space of other devices (in RC mode). This can be achieved through the utilization of the AHB/AXI4-Lite Slave interface or the AXI4 Slave interface (ECAM).

3.12.7.6.3 Access Configuration Space via AHB/AXI4-Lite Slave Interface

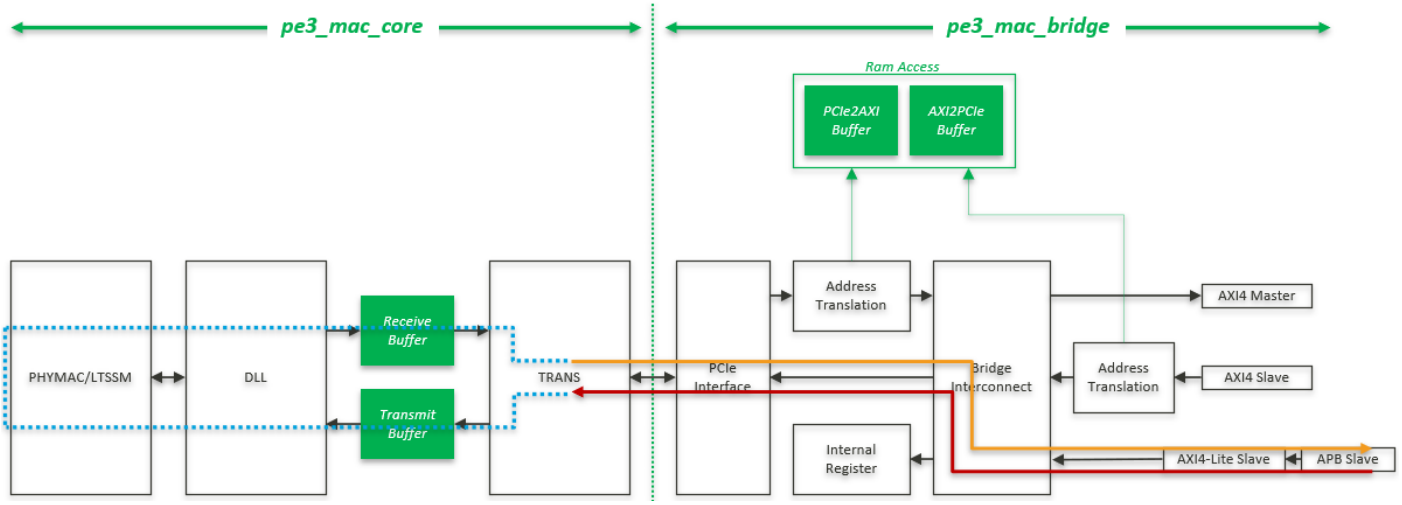


Figure 3-237 RC Configuration Read/Write

In order to access the PCIe controller register within the range of addresses 0x1000 to 0x1FFF, the accesses are routed through the PCIe controller backend configuration space interface. This allows a local processor to effectively read from or write to the PCIe controller configuration space.

Table 3-166 PCIe Configuration Space Register Address for Backend

Byte Address	Read/Write (R/W)	Description
0x1000 – 0x10FF	RO/RW	PCI configuration space
0x1100 – 0x1FFF	RO/RW	PCIe extended configuration space

To access the desired function's configuration Space, it is necessary to appropriately configure the "*PCIE_CFGNUM (0x140)*" register prior to accessing the configuration space. See below for the definitions.

Table 3-167 PCIE_CFGNUM (0x140) Bit Definition

Bit	Definition
Bit [2:0]	FUNC_NUMBER (physical function number)
Bit [7:3]	DEVICE_NUMBER
Bit [15:8]	BUS_NUMBER
Bit [19:16]	BYTE_EN (configuration byte enabled)
Bit [20]	FORCE_BE In situations where the target is the R1C register and the AHB/APB/AXI4-Lite protocol lacks a read strobe, the byte enables of the CFG read or write request should be set to the "BYTE_EN" field value. This will be enforced even in cases where the AHB/APB/AXI4-Lite strobes are absent.
Bit [30:24]	VFUNC_NUMBER (virtual function number)

In the root port mode, the PCIe controller allows the local processor to access bridge configuration Space and send CFG read and write requests via the PCIe interface. To accomplish this, the local processor should configure the *"PCIE_CFGNUM (0x140)"* register accordingly and access the desired register.

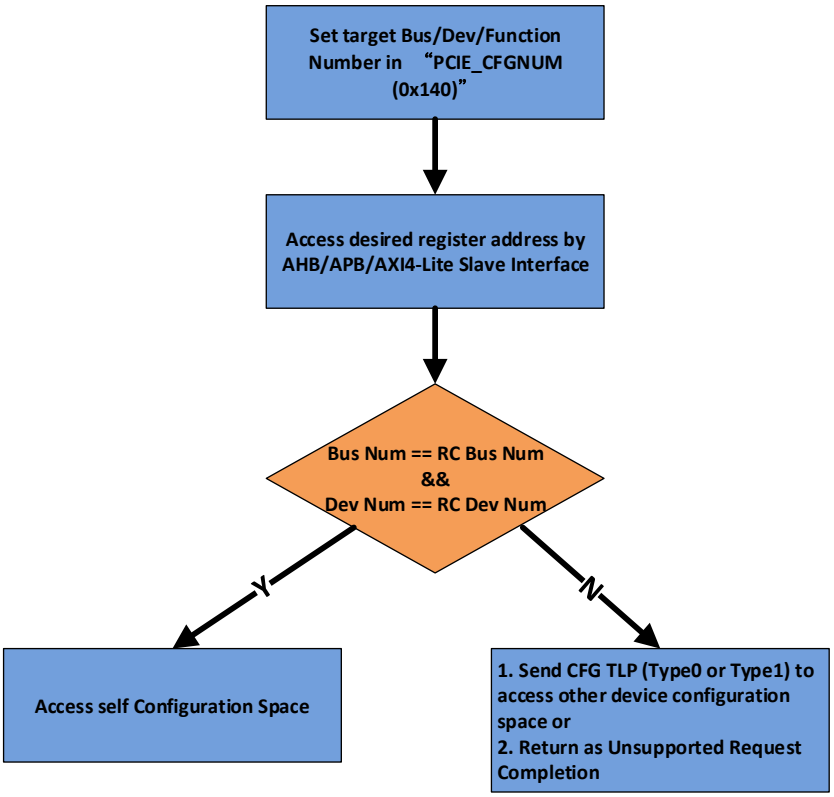


Figure 3-238 RC Access Configuration Space by Backend Interface

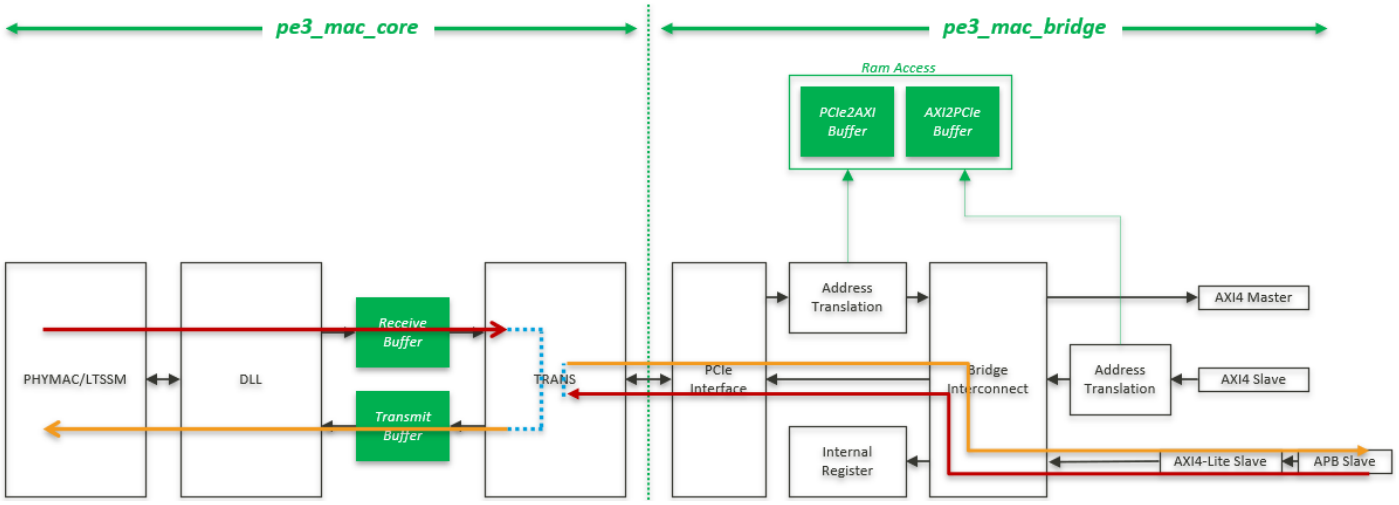


Figure 3-239 EP Configuration Read/Write

When the PCIe controller is in Endpoint Mode and supports multiple functions, the local processor can access Configuration Space via Backend interface by setting "*FUNC_NUMBER*" and "*VFUNC_NUMBER*" in "*PCIE_CFGNUM (0x140)*" to the desired value. Those registers are reserved when the Core is in Endpoint mode and only one function is implemented.

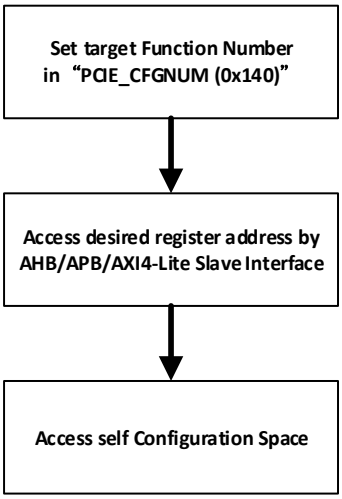


Figure 3-240 EP Access Configuration Space by Backend Interface

3.12.7.6.4 Enhanced Configuration Access Mechanism (ECAM)

When the PCIe controller is set to Root port mode, the local processor on the AXI domain can utilize the Enhanced Configuration Access Mechanism (ECAM) feature to perform CFG read and write requests on the PCIe interface. If ECAM is enabled, the *"PCIE_CFGNUM (0x140)"* register method is no longer applicable.

When the PCIe controller supports ECAM, the AXI slave interfaces can target up to 256MB of PCIe configuration space, rendering the AHB/AXI4-lite slave interface unsuitable for CFG read or write accesses.

To implement ECAM, an address translation table with a table size of 256 Mbytes and translated interface set to PCIe configuration interface should be used. In this case, when a read or write access matches this table, the configuration access is defined by the 28 access LSB:

- [27:20]: Bus Number
- [19:15]: Device Number
- [14:12]: Function Number
- [11:2]: Register Number

There are three scenarios for a configuration request's targeted bus number.

Configuration Request's Targeted Bus Number	Description
Is equal to the root port's primary bus number	• The root port's configuration space is accessed through PCIe backdoor access when the targeted device number is equal to the root port's device number. • Otherwise, an unsupported request completion is returned.
Is equal to the root port's secondary bus number	• A Type 0 configuration request is sent to the PCIe system when the targeted device number is different from 0. • Otherwise, an unsupported request completion is returned.
Falls between the secondary bus number and subordinate bus number	A Type 1 configuration request is sent to the PCIe system.

For other cases, an unsupported request completion is returned.

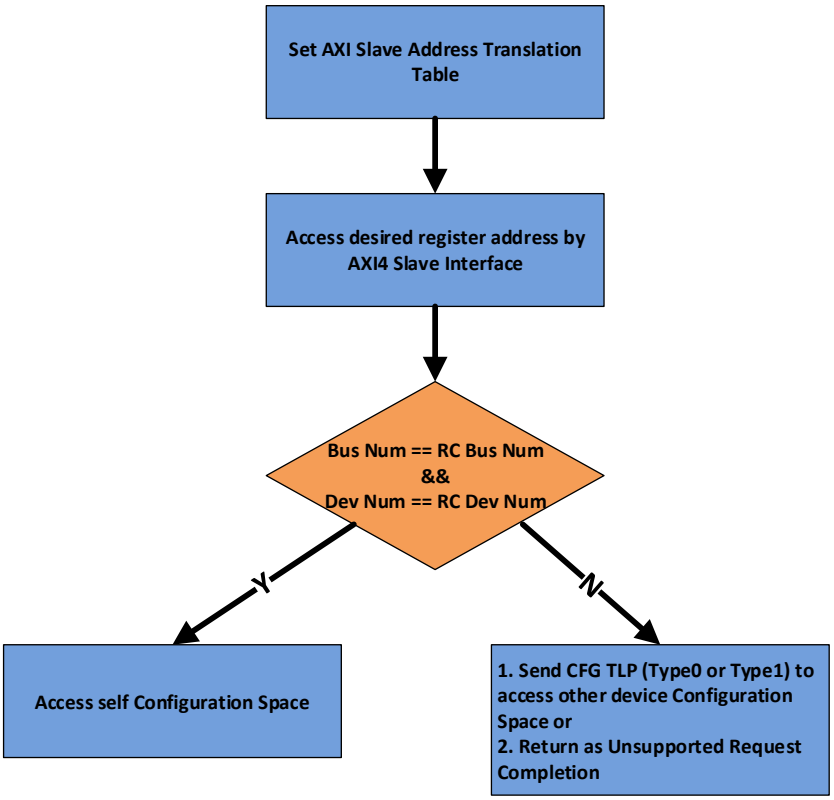


Figure 3-241 RC Access Configuration Space by ECAM

Table 3-168 RC Address Translation Table Setting for ECAM

For AXI slave0 Table0: 0x0800 to 0x081F					
Offset	**Register Name**	**Bit Location**	**Action**	**Value**	**Description**
0x800	ATR_IMPL	[0]	W	1'b1	The table is enabled.
	ATR_SIZE	[6:1]	W	6'd27	Table Size is 2^ (27+1) = 256MB.
	SRC_ADDR_LSB	[31:12]	W	20'b0	SRC_ADDR[ATR_SIZE:0] will be ignored to align the table size.
0x804	SRC_ADDR_MSB	[31:0]	W	32'hBBBBBBBB	Any AXI address in 0xBBBBBBBB_00000000 to 0xBBBBBBBB_0FFFFFFF range will match table0.
0x808	TRSL_ADDR_LSB	[31:12]	W	20'b0	The configuration TLP does not need to set the translated address.
0x80C	TRSL_ADDR_MSB	[31:0]	W	32'h0	The configuration TLP does not need to set the translated address.
0x810	TRSL_ID	[3:0]	W	4'd1	To PCIe configuration interface
	TRSF_PARAM	[18:16]	W	3'd0	If bit [17] is not 1’b1, it is translated to a configuration TLP.
	TRSF_PARAM	[19]	W	1'b0	NW flag is 0.
	TRSF_PARAM	[22:20]	W	3'b0	No snoop
	TRSF_PARAM	[23]	W	1'b0	ECRC is not forwarded.
	TRSF_PARAM	[26:24]	W	3'b0	TC is 0.

3.12.7.7 Programming Guide

Table 3-169 RC Initialization Sequence

Step	Address	Register Name	Local Address	R/W	Value	Description
PCIe Controller Initialization						
1	PCIE_MAC Base address +0x0080	port_type	GEN_SETTINGS[0]	W	1'b1	Port Type must be chosen as Rootport. Please set bit 0 of GEN_SETTINGS register (PCIE_MAC Base Address +0x0080) if the default value is not as expected.
Reset Setting						
2	PCIE_MAC Base address +0x0148	ResetMAC ResetPHY ResetBRG	PCIE_SW [0] PCIE_SW [1] PCIE_SW [2]	W	1'b1 1'b1 1'b1	Software reset PCIE_SW[2:0](PCIE_MAC Base Address +0x148) may be asserted and de-asserted for safety purposes.
3	PCIE_MAC Base address +0x0148	ResetMAC ResetPHY ResetBRG	PCIE_SW [0] PCIE_SW [1] PCIE_SW [2]	W	1'b0 1'b0 1'b0	
4	PCIE_MAC Base address +0x0148	ResetPE	PCIE_SW [3]	W	1'b0	PERST# (PCIE_SW[3] (PCIE_MAC Base Address +0x148)) must be de-asserted for the link sequence. Please refer to the reset scenario description.
Misc. Setting						
5	PCIE_MAC Base address +0x0098	DeviceID VendorID	PCIE_PCI_IDS_0[31:16] PCIE_PCI_IDS_0[15:0]	W	Appropriate value	The Vendor ID, Device ID, Revision ID, Classcode can be set by filling the new value to the PCIE_PCI_IDS_0 register (PCIE_MAC Base Address +0x0098), PCIE_PCI_IDS_1 register (PCIE_MAC Base Address +0x009C) if the default value is not as expected.
6	PCIE_MAC Base address +0x009C	Classcode RevisionID	PCIE_PCI_IDS_1[31:8] PCIE_PCI_IDS_1[7:0]	W	Appropriate value	
Interrupt enable setting						
8	PCIE_MAC Base address +0x0180		IMASK_LOCAL[31:0]	W	Appropriate value	Enable the corresponded interrupt by setting IMASK_LOCAL register (PCIE_MAC Base address+0x0180)
Address Translation setting						
9	PCIE_MAC Base address +0x0600~0x07FF PCIE_MAC Base address	SRC_ADDR_LSB SRC_ADDR_MSB TRSL_ADDR_LSB TRSL_ADDR_MSB TRSL_PARAM	ATR_PCIE_WIN0 ATR_PCIE_WIN1 ATR_AXI_SLV0 ATR_AXI_SLV1 ATR_AXI_SLV2 ATR_AXI_SLV3	W	Appropriate value	Set PCIe Address Translation window for I/O memory, prefetchable and non-prefetchable memory resource. Set the address transaction window to remap PCIe bus

Step	Address	Register Name	Local Address	R/W	Value	Description
	+0x0800~0x0BFF					address if there is demand from the user.
10	PCIE_MAC Base address +0x00FC	PFWIN_64B PFWIN_IMPL IOWIN_32B IOWIN_IMPL	PCIE_BAR_WIN[3:0]	W	Appropriate value	Set 3'b11 to bit 3:2 of PCIE_BAR_WIN (PCIE_MAC Base address+0x00FC) if Prefetchable memory resource is needed. Set 3'b11 to bit 1:0 of PCIE_BAR_WIN (PCIE_MAC Base address+0x00FC) if IO resource is needed.
Configuration space setting						
11	PCIE_MAC Base address +0x1004	IO_Enable MEM_ENABLE Bus_master_enable	PCIE_CONF_HDR1[0] PCIE_CONF_HDR1[1] PCIE_CONF_HDR1[2]	W	1'b1 1'b1 1'b1	Bus_Master_Enable and MEM_ENABLE should be asserted by writing 3'b11 to bit 2:1 of the PCIE_CONF_HDR1 register (PCIE_MAC Base address+0x1004), and set IO Enable by writing 1 to bit 0 of the PCIE_CONF_HDR1 register (PCIE_MAC Base address+0x1004) if IO transaction is needed.
12	PCIE_MAC Base address +0x1010~0x1017	Bar0~Bar1	PCIE_CONF_HDR4 PCIE_CONF_HDR5	W	Appropriate value	The field of BARX register (PCIE_MAC Base address+0x1010~) for RC internal register accessing via PCIe Host should be filled with a proper value. It will be filled up by native driver during Host boot-up.
13	PCIE_MAC Base address +0x1018	Prim_Bus_Num Sec_Bus_Num Sub_Bus_Num	PCIE_CONF_HDR6_TYPE1[7:0] PCIE_CONF_HDR6_TYPE1[15:8] PCIE_CONF_HDR6_TYPE1[23:16]	W	Appropriate value	It is recommended to give an effective Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number by writing appropriate values to bit 23 to 16, bit 15 to 8, and bit 7 to 0 of the PCIE_CONF_HDR6_TYPE1 register (PCIE_MAC Base address+0x1018). It will be filled up by native driver during Host boot-up.

Step	Address	Register Name	Local Address	R/W	Value	Description
14	PCIE_MAC Base address +0x101C PCIE_MAC Base address +0x1030	IO_Base IO_Limit IO_Base_Up IO_Limit_Up	PCIE_CONF_HDR7_TYPE1[7:4] PCIE_CONF_HDR7_TYPE1[15:12] PCIE_CONF_HDR12_TYPE1[15:0] PCIE_CONF_HDR12_TYPE1[31:16]	W	Appropriate value	IO_Base and IO_Limit should be specified according to the downstream device I/O Range to bit 15 to 12 and bit 7 to 4 of the PCIE_CONF_HDR7_TYPE1 register (PCIE_MAC Base address+0x0101C). IO_Base_Up and IO_Limit_Up should also be specified if IOWIN_32B (bit 1 of PCIE_BAR_WIN (PCIE_MAC Base address+0x00FC)) is set. It will be filled up by native driver during Host boot-up.
15	PCIE_MAC Base address +0x1020~0x102F	Mem_Base Mem_Limit Pref_Mem_Base Pref_Mem_Limit Pref_Base_Up_32b Pref_Limit_Up_32b	PCIE_CONF_HDR8_TYPE1[15:4] PCIE_CONF_HDR8_TYPE1[31:20] PCIE_CONF_HDR9_TYPE1[15:4] PCIE_CONF_HDR9_TYPE1[31:20] PCIE_CONF_HDR10_TYPE1 PCIE_CONF_HDR11_TYPE1	W	Appropriate value	Mem_Limit and Mem_Base should be specified according to the downstream device Memory Range to bit 31 to 20 and bit 15 to 4 of the PCIE_CONF_HDR8_TYPE1 register (PCIE_MAC Base address+0x1020). Pref_Mem_Limit and Pref_Mem_Base should also be specified according to the downstream device Memory Range to bit 31 to 20 and bit 15 to 4 of the PCIE_CONF_HDR9_TYPE1 register (PCIE_MAC Base address+0x1024) if PFWIN_IMPL(bit 2 of PCIE_BAR_WIN (PCIE_MAC Base address+0x00FC)) is set. Pref_Base_Up_32 and Pref_Limit_Up_32 should also be specified according to the downstream device Memory Range to PCIE_CONF_HDR10_TYPE1 register (PCIE_MAC Base address+0x1028) and PCIE_CONF_HDR11_TYPE1 register (PCIE_MAC Base address+0x102C) if PFWIN_64B (bit 3 of PCIE_BAR_WIN (PCIE_MAC Base

Step	Address	Register Name	Local Address	R/W	Value	Description
						address+0x00FC)) is set. It will be filled up by native driver during Host boot-up.

Table 3-170 EP Initialization Sequence

Step	Address	Register name	Local address	R/W	Value	Description
PCIe Controller Initialization						
1	PCIE_MAC Base address +0x0080	port_type	GEN_SETTINGS[0]	W	1'b0	Port Type must be chosen as Endpoint. Please set bit 0 of GEN_SETTINGS register (PCIE_MAC Base Address +0x0080) if the default value is not as expected.
Reset Setting						
2	PCIE_MAC Base address +0x0148	ResetMAC ResetPHY ResetBRG	PCIE_SW [0] PCIE_SW [1] PCIE_SW [2]	W	1'b1 1'b1 1'b1	Software reset PCIE_SW [2:0] (PCIE_MAC Base Address +0x148) may be asserted and de-asserted for safety purposes.
3	PCIE_MAC Base address +0x0148	ResetMAC ResetPHY ResetBRG	PCIE_SW [0] PCIE_SW [1] PCIE_SW [2]	W	1'b0 1'b0 1'b0	
Misc. Setting						
4	PCIE_MAC Base address +0x0098	DeviceID VendorID	PCIE_PCI_IDS_0[31:16] PCIE_PCI_IDS_0[15:0]	W	Appropriate value	The Vendor ID, Device ID, Revision ID, Classcode, Subsystem Vendor ID, and Subsystem Device ID can be set by filling in the new value to the PCIE_PCI_IDS_0 register (PCIE_MAC Base Address +0x0098), PCIE_PCI_IDS_1 register (PCIE_MAC Base Address +0x009C), and PCIE_PCI_IDS_2 register (PCIE_MAC Base Address +0x00A0) if the default value is not as expected.
5	PCIE_MAC Base address +0x009C	Classcode RevisionID	PCIE_PCI_IDS_1[31:8] PCIE_PCI_IDS_1[7:0]	W	Appropriate value	
6	PCIE_MAC Base address +0x00A0	Subsystem Device ID Subsystem Vendor ID	PCIE_PCI_IDS_2[31:16] PCIE_PCI_IDS_2[15:0]	W	Appropriate value	
Interrupt Enable Setting						
7	PCIE_MAC Base address +0x0180		IMASK_LOCAL[31:0]	W	Appropriate value	Enable the corresponding interrupt by setting IMASK_LOCAL register (PCIE_MAC Base address+0x0180)
Address Translation Setting						
8	PCIE_MAC Base address +0x0600~0x07FF PCIE_MAC Base address +0x0800~0x0BFF	SRC_ADDR_LSB SRC_ADDR_MSB TRSL_ADDR_LSB TRSL_ADDR_MSB TRSL_PARAM	ATR_PCIE_WIN0 ATR_PCIE_WIN1 ATR_AXI_SLV0 ATR_AXI_SLV1 ATR_AXI_SLV2 ATR_AXI_SLV3	W	Appropriate value	Set PCIe Address Translation window for I/O memory, prefetchable and non-prefetchable memory resource. Set the address transaction window to remap PCIe bus address if there is demand from the user.

Step	Address	Register name	Local address	R/W	Value	Description
Configuration Space Setting						
9	PCIE_MAC Base address +0x1004	IO_Enable MEM_ENABLE Bus_master_enable	PCIE_CONF_HDR1[0] PCIE_CONF_HDR1[1] PCIE_CONF_HDR1[2]	W	1'b1 1'b1 1'b1	Bus_Master_Enable and MEM_ENABLE should be asserted by writing 3'b11 to bit 2:1 of the PCIE_CONF_HDR1 register (PCIE_MAC Base address+0x1004). Set IO Enable by writing 1 to bit 0 of the PCIE_CONF_HDR1 register (PCIE_MAC Base address+0x1004) if IO transaction is needed.
10	PCIE_MAC Base address +0x1010~0x1027	Bar0~Bar5	PCIE_CONF_HDR4~ PCIE_CONF_HDR9	W	Appropriate value	The field of BARx register (PCIE_MAC Base address+0x1010 to 0x1027) for register accessing via PCIe Host should be filled with a proper value. It will be filled up by native driver during Host boot-up.

3.12.7.8 Register Definition

The PCIe module register consists of:

- The Internal Registers of PCIe Controller
- The PCIe Configuration Space and PCIe Extended Configuration Space, accessible through the PCIe Configuration interface
- The Extended Internal Register of PCIe Controller

Table 3-171 PCIe Module Register Mapping

Offset	Description
0x0000 - 0x0FFF	The internal registers of PCIe controller
0x1000 - 0x1FFF	The PCIe configuration space and PCIe extended configuration space, accessible through the PCIe configuration interface
0x2000 - 0x2FFF	Reserved
0x3000 - 0x3FFF	The extended internal registers of PCIe controller
0x4000 - 0x7FFF	Reserved

3.12.7.8.1 Internal Register of PCIe Controller

The internal register of PCIe controller can control PCIe controller. It includes Control, Status, Interrupt, and Event registers. It maps to PCIe module register offset 0x0000 (PCIE_MAC Base address+0x0000).

Table 3-172 PCIe Internal Register

Offset	Description
0x0000 - 0x017F	Control and status registers
0x0180 - 0x01FF	Interrupt and event registers
0x0200 - 0x02FF	Routing, arbitration and priority rules
0x0300 - 0x03FF	Interface optional feature definitions
0x0400 - 0x05FF	DMA engines registers
0x0600 - 0x0BFF	Address translation registers
0x0C00 - 0x0E7F	Optional features register
0x0E80 - 0x0EFF	MSI-X control registers
0x0F00 - 0x0F7F	MSI-X status
0x0F80 - 0x0FFF	MSI-X pending bit array registers
0x1000 - 0x1FFF	Configuration space registers
0x2000 - 0x2FFF	Reserved
0x3000 - 0x3FFF	Reserved
0x4000 - 0x7FFF	MSI-X table registers

3.12.7.8.2 PCIe Configuration Space and PCIe Extended Configuration Space

The PCIe Configuration Space and PCIe Extended Configuration Space are accessible through PCIe Configuration Space interface. The Configuration Space register maps to PCIe module register offset 0x1000 (PCIE_MAC Base address+0x1000).

Table 3-173 PCIe Configuration Space and PCIe Extended Configuration Space

Offset	Description
0x0000 - 0x003F	Type 0/1 standard PCI configuration header
0x0040 - 0x007F	Reserved
0x0080 - 0x00BB	PCIe capability
0x00BC - 0x00CF	Reserved
0x00D0 - 0x00DB	MSI-X capability
0x00DC - 0x00DF	Reserved
0x00E0 - 0x00F7	MSI capability
0x00F8 - 0x00FF	PCI power management capability
0x0100 - 0x0107	Vendor-Specific capability
0x0108 - 0x010F	LTR capability
0x0110 - 0x011F	L1 PM substates capability
0x0120 - 0x01CF	Reserved
0x01D0 - 0x01DB	PTM capability
0x01DC - 0x01FF	Reserved
0x0200 - 0x0247	AER capability
0x0248 - 0x02FF	Reserved
0x0300 - 0x032B	Secondary PCIe extended capability

Offset	Description
0x032C - 0x0FFF	Reserved

3.12.7.9 PCIe PHY

3.12.7.9.1 Overview

The PCIe PHY is responsible for managing the fundamental PCIe protocol and signaling aspects. This includes critical functionalities such as data serialization and de-serialization, 8b/10b encoding/decoding, 128b/130b encoding/decoding (at 16.0 GT/s), analog buffers, elastic buffers, and receiver detection mechanisms. The primary objective of this block is to synchronize the data's clock domain from the original PCIe rate to one that is compatible with the general logic.

3.12.7.9.2 Features

- Direct disparity control for use in transmitting compliance pattern(s)
- Clock and Data Recovery (CDR) from serial stream on the PCIe bus
- 8b/10b encoding/decoding and error indication
- 128b/130b encoding/decoding and error indication
- Port 0 supports RC mode and EP mode
- Port 1 supports RC mode
- Port 0 supports x2 link, link rate of 2.5 GT/s, 5.0 GT/s, and 8.0 GT/s per lane
- Port 1 supports x1 link, link rate of 2.5 GT/s, 5.0 GT/s
- Compliant with PCIe Base Specification Revision 3.0
- Compliant with PHY interface for PCIe (PIPE) 4.0
- Supports legacy PCI power management.
- Supports Active State Power Management (ASPM) L0s and L1 states.
- Supports L1 Power Management Substates (L1PMSS) with CLKREQ#.
- Supports the APB interface for PCIe PHY configuration.

3.12.7.9.3 Block Diagram

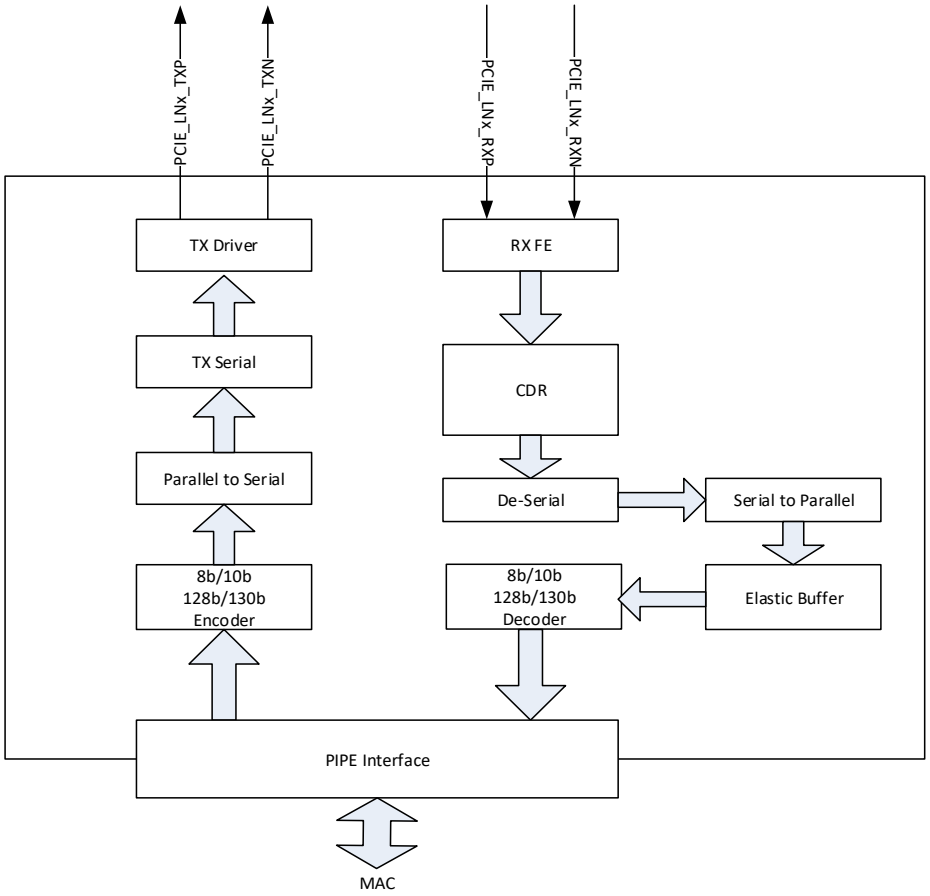


Figure 3-242 Block Diagram of PCIe PHY

Figure 3-242 depicts the block diagram of the PCIe PHY, which comprises two primary sub-modules.

- **Analog PHY (PHYA)**: Contains the TX driver, serializer, RX front-end, CDR, and de-serializer.
- **Digital PHY (PHYD)**: Includes essential features such as 8b/10b encoding/decoding, 128b/130b encoding/decoding (at 16 GT/s), and elastic buffers. These elastic buffers are utilized to account for any differences in frequencies between the bit rates at each end of a link, whereby the PCIe Specification stipulates that the elastic buffer can cover a range of ± 300 ppm.

The interface between the PHY and PCIe controller follows the PIPE specification.

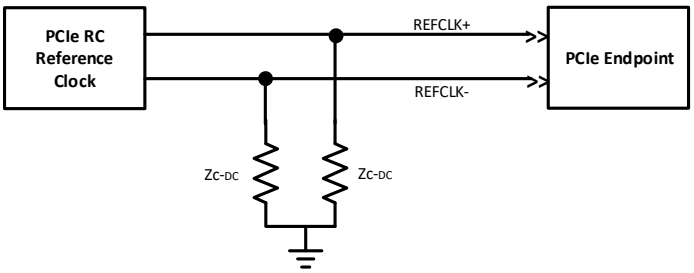


Figure 3-243 PCIe 100 MHz Reference Clock (REFCLK) Architecture on PCB

3.12.7.9.4 Function Description

3.12.7.9.4.1 Far-End Loopback Mode

The request for loopback functionality can be conveyed by setting Symbol 5 bit 2 of the Training Sequence (TS1) ordered set. Upon issuing the loopback request, both the MAC (LTSSM) and PHY must be in Loopback mode. Therefore, the PCIe MAC is responsible for controlling the power-down, Loopback, and TXElecIdle features of the PHY through the PIPE interface. Consequently, the PCIe PHY is set to Loopback mode.

Table 3-174 Loopback Mode Defined in PIPE Specification

powerdown[1:0]	TXDectectRX/Loopback	TXElecIdle	Description
P0: 2'b00	0	0	PHY transmits data. MAC provides data bytes to be sent every clock cycle.
	0	1	PHY does not transmit data and is in electrical idle.
	1	0	PHY goes into loopback mode (Far-end loopback).
	1	1	Illegal. MAC should never do this.

3.12.7.9.4.2 Test Mode for Compliance Test

Upon entering Polling state, the compliance pattern will be transmitted. The Compliance state is designed to evaluate the conformity of the transmitter and interconnect in the device under test setup with the voltage and timing specifications outlined in the PCIe base or CEM standards. To enter polling compliance, one of the following conditions must be met:

- (a) the enter compliance bit in the Link Control 2 register is set to 1'b1 before entering polling active state; or
- (b) a passive test load is applied to all transmitter lanes.

3.12.7.9.5 PCIe Electrical Characteristics

The PCIe electrical characteristics conform to the standards specified in the *PCIe Base Specification Revision 4.0* and *PCIe Card Electromechanical Specification Revision 4.0* for system board usages.

Table 3-175 PCIe 2.5 GT/s Electrical Characteristics

Description	Min	Typ	Max	Unit
Data rate		2.5		GT/s
Unit interval(1)	399.88	-	400.12	ps
Transmitter parameters				
TX differential peak-to-peak output voltage(2)	253	-	1200	mV
TX eye width(2)	246	-	-	ps
TX DC differential impedance	80	-	120	Ω
Receiver parameters				
RX eye width(3)	287	-	-	ps
RX DC differential impedance	80	-	120	Ω

(1) Do not account for SSC caused variations.

(2) Refer to *PCI Express Card Electromechanical Specification Revision 4.0, Section 4.8.10, Table 24*.

(3) Refer to *PCI Express Card Electromechanical Specification Revision 4.0, Section 4.8.1, Table 29*.

Table 3-176 PCIe 5.0 GT/s Electrical Characteristics

Description	Min	Typ	Max	Unit
Data rate		5.0		GT/s
Unit interval(1)	199.94	-	200.06	ps
Transmitter parameters				
TX differential peak-to-peak output voltage(2)	225	-	1200	mV
TX eye width(2)	95	-	-	ps
TX DC differential impedance	-	-	120	Ω
Receiver parameters				
1.5 to 100 MHz RMS jitter(3)	1.4	-	-	ps RMS
< 1.5 MHz RMS Jitter	3.0			ps RMS
1.5 – 100 MHz Dj	30			ps PP
> 100 MHz Dj	27			ps PP

(1) Do not account for SSC caused variations.

(2) Refer to *PCIe Card Electromechanical Specification Revision 3.0, Section 4.8.2, Table 4-7.*

(3) Refer to *PCIe Card Electromechanical Specification Revision 3.0, Section 4.8.5, Table 4-13.*

Table 3-177 PCIe 8.0 GT/s Electrical Characteristics

Description	Min	Typ	Max	Unit
Data rate		8.0		GT/s
Unit interval(1)	124.9625	-	125.0375	ps
Transmitter parameters				
TX differential peak-to-peak output voltage(2)	34	-	1300	mV
TX eye width(2)	41.25	-	-	ps
TX DC differential impedance	-	-	120	Ω
Receiver parameters				
RX sinusoidal jitter at 100 MHz(3)	12.5	-	-	ps PP

(1) Do not account for SSC caused variations.

(2) Refer to *PCI Express Card Electromechanical Specification Revision 4.0, Section 4.8.12, Table 27*.

(3) Refer to *PCI Express Card Electromechanical Specification Revision 4.0, Section 4.8.16, Table 32*.

Table 3-178 PCI Express 16.0 GT/s Electrical Characteristics

Description	Min	Typ	Max	Unit
Data rate		16.0		GT/s
Unit interval(1)	62.48125	-	62.51875	ps
Transmitter parameters				
TX differential peak-to-peak output voltage(2)	19	-	1300	mV
TX eye width(2)	21.75	-	-	ps
TX DC differential impedance	-	-	120	Ω
Receiver parameters				
RX sinusoidal jitter at 100 MHz(3)	6.25	-	-	ps PP

(1) Do not account for SSC caused variations.

(2) Refer to *PCI Express Card Electromechanical Specification Revision 4.0, Section 4.8.13, Table 28*.

(3) Refer to *PCI Express Card Electromechanical Specification Revision 4.0, Section 4.8.17, Table 33*.

Table 3-179 PCIe 100 MHz REFCLK Electrical Characteristics

Description	Min	Typ	Max	Unit
Rising Edge Rate(1)	0.6		4	V/ns
Falling Edge Rate	0.6		4	V/ns
Differential Input High Voltage	+150			mV
Differential Input Low Voltage			-150	mV
Absolute Crossing Point Voltage (V_{CROSS})	+250		+550	mV
Variation of V_{CROSS} over all rising clock edges			+140	mV
Ring-back Voltage Margin (V_{RB})	-100		+100	mV
Time before V_{RB} is allowed	500			ps
Average Clock Period Accuracy	-300		+2800	ppm
Absolute Period (including Jitter and Spread Spectrum Modulation)	9.847		10.203	ns
Cycle to Cycle jitter			150	ps
Absolute Max Input Voltage			1.15	V
Absolute Min Input Voltage	-0.3			V
Duty Cycle	40		60	%
Rising edge rate (REFCLK+) to falling edge rate (REFCLK-) matching			20	%
Clock source DC impedance	40		60	Ω

(1) Before application of SSC.

3.12.7.9.6 Power Management

Table 3-180 Power Requirement

Description	Min	Typ	Max	Unit
AVDD12(1)	1.2V-7%	1.2	1.2V+5%	V
AVDD15(1)	1.5V-7%	1.5	1.5V+5%	V

(1) The power voltage is defined in package ball node and tolerance includes PCB IR and PMIC variation (DC 1% + AC 4%).

3.12.7.10 References

- PCIe Base Specification Revision 3.0
- PCIe Card Electromechanical Specification Revision 3.0
- PCI Power Management Specification Version 1.2
- PHY Interface For the PCIe Version 3.0
- AMBA AXI Specification Version 2.0
- AMBA 4 AXI4 Stream Protocol Specification, Version 1.0 - ARM, March 2010

3.12.7.11 Appendix

3.12.7.11.1 Clock Related Signal

Signal	I/O	Width	Description	Remark
pl_pclk	I	1	PIPE clock from PHY.	250 MHz
tl_clk/br_clk	I	1	PCIe Transaction Layer and Bridge Layer Clock from SYSTEM. The two clocks should belong to the same clock group.	
axi_clk250	I	1	AXI clock from the AXI Bus. The clock can be the same as “tl_clk” if AXI runs in “tl_clk” domain.	
ref_clk	I	1	PCIe timer clock from the system. The clock can be the same as “tl_clk” if “tl_clk” is fixed frequency.	>= 20 MHz
ahb_apb_clk	I	1	AHB/APB interface clock from AHB/APB. The clock can be the same as “tl_clk” if AHB/APB interface runs in “tl_clk” domain.	
mbist_diag_clk	I	1	MBIST diagnosis clock. If there is no diagnosis circuit in MAC, tie to 0.	
pcie_ref_clock_freq	I	22	Indicates the “ref_clk” frequency. For example, if ref_clk is 26 MHz, tie to fixed value 26 or register with default value 26.	
pcie_tl_clock_freq	I	22	Indicates the “tl_clk” frequency. For example, if tl_clk is 200 MHz, tie to fixed value 200 or register with default value 200.	
scan_clk	I	1	DFT related signal If the DFT flow is not expected to initiate currently, tie to 0.	

3.12.7.11.2 AHB Slave Interface

Signal	I/O	Width	Description
ahb_pcie_hsel	I	1	Each AHB slave has its own slave select signal and this signal indicates that the current transfer is intended for the selected slave.
ahb_pcie_htrans	I	2	Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE or BUSY.
ahb_pcie_haddr	I	32	The 32-bit system address bus.
ahb_pcie_hwrite	I	1	When HIGH, this signal indicates a write transfer and when LOW, a read transfer.
ahb_pcie_hsize	I	2	Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit) or word (32-bit).
ahb_pcie_hwdata	I	32	The write data bus is used to transfer data from the master to the bus slaves during write operations.
ahb_pcie_hbstrb	I	4	Indicates which byte in HWDATA will be written in a write transfer. If AHB master does not support write strobe, just tie the signal to 4'hf.
ahb_pcie_hready	I	1	When HIGH, the HREADY_IN signal informs all slaves that the previous transfer is complete.

Signal	I/O	Width	Description
ahb_pcie_slv_hrdata	O	32	The read data bus is used to transfer data from bus slaves to the bus master during read operations.
ahb_pcie_slv_hready	O	1	When HIGH, the HREADY signal indicates that a transfer has finished on the bus.

3.12.7.11.3 AXI Master Interface

The table below enumerates the signals present in the AXI4 master 0 interface. These signals are also utilized for the following interface signals:

- AXI4 Master 1 (axi4_mst1_)
- AXI4 Master 2 (axi4_mst2_)
- AXI4 Master 3 (axi4_mst3_)

Signal	Width	I/O	Description
axi4_mst0_awid	4	O	Write address ID
axi4_mst0_awaddr	64	O	Write address bus
axi4_mst0_awregion	4	O	Write region
axi4_mst0_awlen	8	O	Burst length
axi4_mst0_awsize	3	O	Burst size
axi4_mst0_awburst	2	O	Burst type
axi4_mst0_awlock	1	O	Lock type
axi4_mst0_awcache	4	O	Cache type
axi4_mst0_awprot	3	O	Protection type
axi4_mst0_awqos	4	O	QoS value
axi4_mst0_awfunc	12	O	Function Number: When several physical and/or virtual functions are implemented, this signal indicates which function has initiated the request. • Bit [8:0]: Virtual Function Number: – 0 indicates that the physical function has initiated the request, – 1 - 511 indicate that one of the virtual functions 1 to 511 has initiated the request. This signal must be tied to 0s if virtual functions are not implemented. • Bit [11:9]: Physical Function Number This signal must be tied to 0s if only one physical function is implemented.
axi4_mst0_awuser	32	O	Write Address user side band signal: • Bit [0]: Asserted to 1b when an ECRC error is detected in the AXI Burst • Bit [1]: Asserted to 1b when a PCIe Memory Error (Poisoned TLP Reception or Receive Buffer Memory Error) is detected in the AXI Burst • Bit [2]: Asserted to 1b when a Bridge Buffer Memory Error is detected in the AXI Burst

Signal	Width	I/O	Description
			• Bit [3]: Asserted to 1b when an AXI Application Data Error is detected in the AXI Burst (only relevant if the transaction's requester is an AXI slave Interface). • Bit [7:4]: Provides the transaction's requester module. Allowed values for this field are: – 4'd0: PCIe RX Interface, received on even BAR – 4'd1: PCIe RX Interface, received on odd BAR – 4'd2: AXI4-Lite Slave Interface – 4'd4 + k: AXI4 Slave Number k=0 - 3 Interface – 4'd8 + k: DMA Engine Number k=0 - 7 • Bit [9:8]: Address Type (00b: Untranslated, 01b: Translation Request, 10b: Translated) • Bit [10]: Reserved • Bit [12:11]: When the transaction's requester module is the PCIe RX Interface, this field indicates from which BAR the request has come. Allowed values for this field are: – 2'd0: Request was received on BAR0 when AXI4_MST0_AWUSER[4]=0, or on BAR1 when AXI4_MST0_AWUSER[4]=1 – 2'd1: Request was received on BAR2 when AXI4_MST0_AWUSER[4]=0, or on BAR3 when AXI4_MST0_AWUSER[4]=1 – 2'd2: Request was received on BAR4 when AXI4_MST0_AWUSER[4]=0, or on BAR5 when AXI4_MST0_AWUSER[4]=1 – 2'd3: Request was received on expansion ROM • Bit [13]: PCIe attribute value for No Snoop • Bit [14]: PCIe attribute value for relaxed ordering • Bit [15]: PCIe attribute value for ID-based ordering • Bit [31:16]: PCIe requester ID
axi4_mst0_awvalid	1	I	Write address valid
axi4_mst0_awready	1	O	Write address ready
axi4_mst0_wid	4	O	Write ID (for AXI3; unused in AXI4)
axi4_mst0_wdata	64, 128, or 256	O	Write ID (for AXI3; unused in AXI4)
axi4_mst0_wderr	8, 16, or 32	O	Write Data Error: • If G_DATA_PROT=0: – Bit [0]: Bridge Buffer uncorrectable read error: this bit is asserted when an uncorrectable error has been detected by the memory's ECC logic when reading data from the PCIe2AXI buffer. The error is reported on the same clock cycle as the affected data. – Bit [31:1]: Reserved • If G_DATA_PROT=1: – Bit [0]: Parity for AXI4_MST0_WDATA[31:0]

Signal	Width	I/O	Description
			– Bit [3:1]: Reserved – Bit [4]: Parity for AXI4_MST0_WDATA[63:32] – Bit [7:5]: Reserved – … – Bit [28]: Parity for AXI4_MST0_WDATA[255:224] – Bit [31:29]: Reserved • If G_DATA_PROT=4: – Bit [0]: Parity for AXI4_MST0_WDATA[7:0] – Bit [0]: Parity for AXI4_MST0_WDATA[15:8] – … – Bit [31]: Parity for AXI4_MST0_WDATA[255:248] When data protection is implemented (G_DATA_PROT =1 or =4), this signal contains the data protection bits for the data transmitted on AXI4_MST0_WDATA on the same clock cycle. Note that parity is not checked for 8-bit bytes or 32-bit words that are not valid, as indicated by AXI4_MST0_WSTRB on the same clock cycle. Note also that bridge buffer uncorrectable read errors are reported by corrupting all data protection bits on the same clock cycle as the affected data.
axi4_mst0_wstrb	8, 16, or 32	O	Write strobes
axi4_mst0_wlast	1	O	Write last
axi4_mst0_wvalid	1	O	Write valid
axi4_mst0_wready	1	I	Write ready
axi4_mst0_bid	4	I	Response ID
axi4_mst0_bresp	2	I	Write response
axi4_mst0_bvalid	1	I	Write response valid
axi4_mst0_bready	1	O	Write response
axi4_mst0_arid	4	O	Read address ID
axi4_mst0_araddr	64	O	Read address bus
axi4_mst0_arregion	4	O	Read region
axi4_mst0_arlen	8	O	Burst length
axi4_mst0_arsize	3	O	Burst size
axi4_mst0_arburst	2	O	Burst type
axi4_mst0_arlock	1	O	Lock type
axi4_mst0_arcache	4	O	Cache type
axi4_mst0_arprot	3	O	Protection type
axi4_mst0_arqos	4	O	QoS value
axi4_mst0_arfunc	12	O	Function Number: When several physical and/or virtual functions are implemented, this signal indicates which function has initiated the request. • Bit [8:0]: Virtual Function Number: – 0 indicates that the physical function has initiated the request,

Signal	Width	I/O	Description
			– 1 - 511 indicate that one of the virtual functions 1 to 511 has initiated the request. This signal must be tied to 0s if virtual functions are not implemented. • Bit [11:9]: Physical Function Number This signal must be tied to 0s if only one physical function is implemented.
axi4_mst0_aruser	32	O	Read Address user side band signal: • Bit [0]: Asserted to 1b when an ECRC error is detected in the AXI Burst • Bit [1]: Asserted to 1b when a PCIe Memory Error (Poisoned TLP Reception or Receive Buffer Memory Error) is detected in the AXI Burst • Bit [2]: Asserted to 1b when a Bridge Buffer Memory Error is detected in the AXI Burst. • Bit [3]: Asserted to 1b when an AXI Application Data Error is detected in the AXI Burst (only relevant if the transaction's requester is an AXI slave Interface). • Bit [7:4]: Provides the transaction's requester module. Allowed values for this field are: – 4'd0: PCIe RX Interface, received on even BAR – 4'd1: PCIe RX Interface, received on odd BAR – 4'd2: AXI4-Lite Slave Interface – 4'd4 + k: AXI4 Slave Number k=0 - 3 Interface – 4'd8 + k: DMA Engine Number k=0.7 • Bit [9:8]: Address Type (00b: Untranslated, 01b: Translation Request, 10b: Translated) • Bit [10]: Translation Request No Write (NW) flag • Bit [12:11]: When the transaction's requester module is PCIe RX Interface, this field indicates from which BAR the request has come. Allowed values for this field are: – 2'd0: Request was received on BAR0 when AXI4_MST0_ARUSER[4]=0, or on BAR1 when AXI4_MST0_ARUSER[4]=1 – 2'd1: Request was received on BAR2 when AXI4_MST0_ARUSER[4]=0, or on BAR3 when AXI4_MST0_ARUSER[4]=1 – 2'd2: Request was received on BAR4 when AXI4_MST0_ARUSER[4], or on BAR5 when AXI4_MST0_ARUSER[4]=1 – 2'd3: Request was received on Expansion ROM • Bit [13]: PCIe Attribute value for No Snoop • Bit [14]: PCIe Attribute value for Relaxed Ordering • Bit [15]: PCIe Attribute value for ID-Based Ordering • Bit [31:16]: PCIe Requester ID
axi4_mst0_arvalid	1	O	Read address valid

Signal	Width	I/O	Description
axi4_mst0_arready	1	I	Read address ready
axi4_mst0_rid	4	I	Response ID
axi4_mst0_rdata	64, 128, or 256	I	Read data
axi4_mst0_rderr	8, 16, or 32	I	Read Data Error: • If G_DATA_PROT=0: – Bit [0]: Buffer uncorrectable read error: this bit allows the application to report a data error in the data phase(s) in which it is asserted. – Bit [31:1]: Reserved. • If G_DATA_PROT=1: – Bit [0]: Parity for AXI4_MST0_RDATA[31:0] – Bit [3:1]: Reserved – Bit [4]: Parity for AXI4_MST0_RDATA[63:32] – Bit [7:5]: Reserved – ... – Bit [28]: Parity for AXI4_MST0_RDATA[255:224] – Bit [31:29]: Reserved • If G_DATA_PROT=4: – Bit [0]: Parity for AXI4_MST0_RDATA[7:0] – Bit [0]: Parity for AXI4_MST0_RDATA[15:8] – ... – Bit [31]: Parity for AXI4_MST0_RDATA[255:248] When data protection is implemented (G_DATA_PROT =1 or =4), this signal contains the data protection bits for the data transmitted on AXI4_MST0_RDATA on the same clock cycle. Note that parity is not checked for 8-bit bytes or 32-bit words that are not valid, as indicated by AXI4_MST0_ARADDR/ARSIZE/ARLEN on the same clock cycle.
axi4_mst0_rresp	2	I	Read response
axi4_mst0_rlast	1	I	Read last
axi4_mst0_ruser	32	I	Read user side band signal: • Bit [3:0]: Reserved • Bit [4]: Asserted to 1b to indicate that this burst must be discarded. If the device that initiated the read request is a PCIe device, then this device will later detect a PCIe Completion timeout. • Bit [31:5]: Reserved
axi4_mst0_rvalid	1	I	Read valid
axi4_mst0_rready	1	O	Read ready

3.12.7.11.4 AXI4 Slave Interfaces

The following table lists the signals in the AXI4 Slave 0 interface. The same signals are used for the AXI4 Slave 1 (axi4_slv1_), AXI4 Slave 2 (axi4_slv2_), and AXI4 Slave 3 (axi4_slv3_) interface signals.

Signal	Width	I/O	Description
axi4_slv0_awid	4	I	Write address ID
axi4_slv0_awaddr	64	I	Write address bus
axi4_slv0_awregion	4	I	Write region
axi4_slv0_awlen	8	I	Burst length
axi4_slv0_awsize	3	I	Burst size
axi4_slv0_awburst	2	I	Burst type
axi4_slv0_awlock	1	I	Lock type
axi4_slv0_awcache	4	I	Cache type
axi4_slv0_awprot	3	I	Protection type
axi4_slv0_awqos	4	I	QoS value
axi4_slv0_awfunc	12	I	Function Number: When several physical and/or virtual functions are implemented, this signal indicates which function has initiated the request. • Bit [8:0]: Virtual Function Number: – 0 indicates that the physical function has initiated the request, – 1 - 511 indicate that one of the virtual functions 1 to 511 has initiated the request. This signal must be tied to 0s if virtual functions are not implemented. • Bit [11:9]: Physical Function Number This signal must be tied to 0s if only one physical function is implemented.
axi4_slv0_awvalid	1	I	Write address valid
axi4_slv0_awready	1	O	Write address ready
axi4_slv0_wid	4	I	Write ID (for AXI3; unused in AXI4)
axi4_slv0_wdata	64, 128, or 256	I	Write data
axi4_slv0_wderr	8, 16, or 32	I	Write Data Error: • If G_DATA_PROT=0: – Bit [0]: Buffer uncorrectable read error: this bit allows the application to report a data error in the data phase(s) in which it is asserted. – Bit [31:1]: Reserved. • If G_DATA_PROT=1: – Bit [0]: Parity for AXI4_SLV0_WDATA[31:0] – Bit [3:1]: Reserved – Bit [4]: Parity for AXI4_SLV0_WDATA[63:32] – Bit [7:5]: Reserved – ... – Bit [28]: Parity for AXI4_SLV0_WDATA[255:224] – Bit [31:29]: Reserved • If G_DATA_PROT=4: – Bit [0]: Parity for AXI4_SLV0_WDATA[7:0] – Bit [0]: Parity for AXI4_SLV0_WDATA[15:8]

Signal	Width	I/O	Description
			– ... – Bit [31]: Parity for AXI4_SLV0_WDATA[255:248] When data protection is implemented (G_DATA_PROT =1 or =4), this signal contains the data protection bits for the data transmitted on AXI4_SLV0_WDATA on the same clock cycle. Note that parity is not checked for 8-bit bytes or 32-bit words that are not valid, as indicated by AXI4_SLV0_WSTRB on the same clock cycle.
axi4_slv0_wstrb	8, 16, or 32	I	Write strobes
axi4_slv0_wlast	1	I	Write last
axi4_slv0_wvalid	1	I	Write valid
axi4_slv0_wready	1	O	Write ready
axi4_slv0_bid	4	O	Response ID
axi4_slv0_bresp	2	O	Write response
axi4_slv0_bvalid	1	O	Write response valid
axi4_slv0_bready	1	I	Response ready
axi4_slv0_arid	4	I	Read address ID
axi4_slv0_araddr	64	I	Read address bus
axi4_slv0_arregion	4	I	Read region
axi4_slv0_arlen	8	I	Burst length
axi4_slv0_arsize	3	I	Burst size
axi4_slv0_arburst	2	I	Burst type
axi4_slv0_arlock	1	I	Lock type
axi4_slv0_arcache	4	I	Cache type
axi4_slv0_arprot	3	I	Protection type
axi4_slv0_arqos	4	I	QoS value
axi4_slv0_arfunc	12	I	Function Number: When several physical and/or virtual functions are implemented, this signal indicates which function has initiated the request. • Bit [8:0]: Virtual Function Number: – 0 indicates that the physical function has initiated the request – 1 - 511 indicate that one of the virtual functions 1 to 511 has initiated the request. This signal must be tied to 0s if virtual functions are not implemented. • Bit [11:9]: Physical Function Number This signal must be tied to 0s if only one physical function is implemented.
axi4_slv0_arvalid	1	I	Read address valid
axi4_slv0_arready	1	O	Read address ready
axi4_slv0_rid	4	O	Response ID
axi4_slv0_rdata	64, 128, or 256	O	Read data
axi4_slv0_rderr	8, 16, or 32	O	Read Data Error: • If G_DATA_PROT=0:

Signal	Width	I/O	Description
			– Bit [0]: Bridge Buffer uncorrectable read error: this bit is asserted when an uncorrectable error has been detected by the memory's ECC logic when reading data from the PCIe2AXI buffer. The error is reported on the same clock cycle as the affected data. – Bit [31:1]: Reserved. • If G_DATA_PROT=1: – Bit [0]: Parity for AXI4_SLV0_RDATA[31:0] – Bit [3:1]: Reserved – Bit [4]: Parity for AXI4_SLV0_RDATA[63:32] – Bit [7:5]: Reserved – ... – Bit [28]: Parity for AXI4_SLV0_RDATA[255:224] – Bit [31:29]: Reserved • If G_DATA_PROT=4: – Bit [0]: Parity for AXI4_SLV0_RDATA[7:0] – Bit [0]: Parity for AXI4_SLV0_RDATA[15:8] – ... – Bit [31]: Parity for AXI4_SLV0_RDATA[255:248] When data protection is implemented (G_DATA_PROT = 1 or = 4), this signal contains the data protection bits for the data transmitted on AXI4_SLV0_RDATA on the same clock cycle. Parity is not checked for 8-bit bytes or 32-bit words that are not valid, as indicated by AXI4_SLV0_ARADDR/ARSIZE/ARLEN. Bridge Buffer uncorrectable read errors are reported by corrupting all data protection bits on the same clock cycle as the affected data.
axi4_slv0_rresp	2	O	Read response
axi4_slv0_rlast	1	O	Read last
axi4_slv0_ruser	32	O	Read user side band signal: • Bit [0]: Asserted to 1b when an ECRC error is detected in the AXI Burst • Bit [1]: Asserted to 1b when a PCIe Memory Error (Poisoned TLP Reception or Receive Buffer Memory Error) is detected in the AXI Burst • Bit [2]: Asserted to 1b when a Bridge Buffer Memory Error is detected in the AXI Burst • Bit [3]: Reserved • Bit [4]: Asserted to 1b to indicate that this AXI Burst was inferred by the Bridge due to a PCIe completion timeout or a discarded AXI packet (with the read response = SLVERR) • Bit [12:5]: Reserved • Bit [13]: PCIe Attribute value for No Snoop • Bit [14]: PCIe Attribute value for Relaxed Ordering • Bit [15]: PCIe Attribute value for ID-Based Ordering • Bit [31:16]: PCIe Completer ID

Signal	Width	I/O	Description
axi4_slv0_rvalid	1	O	Read valid
axi4_slv0_rready	1	I	Read ready

3.12.7.11.5 PIPE Interface

Signal	I/O	Width	Description
pl_powerdown (common to all lanes)	O	2	Power Down: determines the power state (P0, P0s, P1, or P2).
pl_rate (common to all lanes)	O	2	Link Signaling Rate: controls the link signaling rate: • 00: Use 2.5 GT/s signaling rate • 01: Use 5.0 GT/s signaling rate • 10: Use 8.0 GT/s signaling rate • 11: Reserved
pl_width (common to all lanes)	O	2	PIPE Interface Width: indicates current PIPE interface width • 00: 8-bit • 01: 16-bit • 10: 32-bit • 11: 64-bit This signal can be left unconnected if not present on the PHY
pl_pclk_rate (common to all lanes)	O	3	PIPE Interface Clock Rate: indicates current PIPE interface clock rate • 000: 62.5 MHz • 001: 125 MHz • 010: 250 MHz • 011: 500 MHz • 100: 1 GHz • 111: 101: Reserved This signal can be left unconnected if not present on the PHY.
pl_txmargin (common to all lanes)	O	3	Transmit Margin: selects transmitter voltage levels.
pl_txswing (common to all lanes)	O	1	Transmitter Voltage Swing Level
pl_txdeemph (common to all lanes)	O	2	Transmitter Deemphasis at 5.0 GT/s Only for Gen2 PHY.
pl_blockaligncontrol (common to all lanes)	O	1	Block Alignment Control: Used at 8.0 GT/s or above only. This signal can be left unconnected if this port does not exist on the PHY.
pl_txdetectrx_common (common to all lanes)	O	1	Transmit Detect Receive: Prompts the PHY to start a receiver detection operation or to begin loopback. If the PHY has a single TX/RX Detect input, please use the signal.

Signal	I/O	Width	Description
pl_l1ss_l1ss_en (common to all lanes)	O	1	For L1SS Function. Enable L1SS function in PHY. Proprietary signal for MediaTek PHY. This signal can be left unconnected if not present on the PHY.
pl_l1ss_rx_ei_dis (common to all lanes)	O	1	For L1SS Function. Disable electrical idle detect logic in PHY. Proprietary signal for MediaTek PHY. This signal can be left unconnected if not present on the PHY.
pl_l1ss_tx_cm_dis (common to all lanes)	O	1	For L1SS Function. Disable transmitter DC common mode logic in PHY. Proprietary signal for MediaTek PHY. This signal can be left unconnected if not present on the PHY.
pl_l1ss_l1p2_prd (common to all lanes)	O	1	For L1SS Function. Indicate MAC is in L1.2 state including L1.2_ETNRY/ L1.2_IDLE/ L1.2_EXIT Proprietary signal for MediaTek PHY. This signal can be left unconnected if not present on the PHY.
pl_sris_enable (common to all lanes)	O	1	For SRIS Function. Enable SRIS function in PHY. This signal can be left unconnected if not present on the PHY.
pl_txdetectrx (per lane)	O	N_L	Transmit Detect Receive: Prompts the PHY to start a receiver detection operation or to begin loopback. If the PHY has per-lane TX-Detect-RX input, please use the signal.
pl_txdata (per lane)	O	G_PCIE_PIPE_IF _W*8*N_L	Transmit Data
pl_txdatak (per lane)	O	G_PCIE_PIPE_IF _W_G2*N_L	Transmit Data Control The DATAK width depends on Gen2 data width.
pl_txdatavalid (per lane)	O	N_L	Used at 8.0 GT/s or above only
pl_txstartblock (per lane)	O	N_L	Used at 8.0 GT/s or above only
pl_txsyncheader (per lane)	O	2*N_L	Used at 8.0 GT/s or above only
pl_txelecidle (per lane)	O	N_L	Transmit Electrical Idle: Forces the transmit output to electrical idle.
pl_txcompliance	O	N_L	Transmit Compliance:

Signal	I/O	Width	Description
(per lane)			Forces the running disparity to negative in Compliance mode (negative COM character) and can also be used to turn off Lanes that are not initialized
pl_rxpolarity (per lane)	O	N_L	Receive Polarity: Prompts the PHY layer to perform a polarity inversion on the receiver decoding block.
pl_rxstandby (per lane)	O	N_L	PHY RX Control. This signal can be left unconnected if this port does not exist on the PHY. This signal can be left unconnected if not present on the PHY.
pl_m2p_msgbus (per lane)	O	8*N_L	Message bus to PHY If supported speed of PHY is under Gen4, just left unconnected.
pl_phystatus (per lane)	I	N_L	PHY Status: If PHY has a single PHY status output, the same value can be replicated on all PL_PHYSTATUS bits.
pl_rxstatus (per lane)	I	3*N_L	Receive Status: encodes receive status and error codes for the receive data stream and receiver detection.
pl_rxdata (per lane)	I	G_PCIE_PIPE_IF _W*8*N_L	Receive Data
pl_rxdatak (per lane)	I	G_PCIE_PIPE_IF _W_G2*N_L	Receive Data Control The DATAK width depends on Gen2 data width.
pl_rxdatavalid (per lane)	I	N_L	Used at 8.0 GT/s or above only
pl_rxstartblock (per lane)	I	N_L	Used at 8.0 GT/s or above only
pl_rxsyncheader (per lane)	I	2*N_L	Used at 8.0 GT/s or above only
pl_rxelecidle (per lane)	I	N_L	Receive Electrical Idle: indicates-that electrical idle is detected on receiver lane.
pl_p2m_msgbus (per lane)	I	8*N_L	Message bus from PHY If supported speed of PHY is under Gen4, just tie 0.
pl_eq_rxeqinprogress_common (common to all lanes)	O	2	EQ Action when link speed changes to Gen3 or Gen4. 00: Idle 01: Do EQ 10: SKIP EQ 11: Reset Proprietary signal for MediaTek PHY. This signal can be left unconnected if not present on the PHY.
pl_eq_rxeqeval_common (common to all lanes)	O	1	RX Evaluation request. If the PHY has a single RX Evaluation input, please use the signal

Signal	I/O	Width	Description
pl_eq_getlocalpresetcoefficients (per lane)	O	N_L	These signals are used to query coefficients corresponding to a preset from the PHY. If this interface is not present on then PHY, then inputs can be tied to 0's and outputs can be left unconnected. Note: If bit 4 of PIPE_LOCALPRESETINDEX is not present on the PHY, then it can be left unconnected.
pl_eq_localtxcoefficientsvalid (per lane)	I	N_L	
pl_eq_localpresetindex (per lane)	O	5*N_L	
pl_eq_localtxpresetcoefficients (per lane)	I	18*N_L	
pl_eq_localfs (per lane)	I	6*N_L	FS setting from PHY.
pl_eq_locallf (per lane)	I	6*N_L	LF setting from PHY.
pl_eq_rxeqeval (per lane)	O	N_L	RX Evaluation request. If the PHY has per-lane RX Evaluation input, please use the signal.
pl_eq_rxeqinprogress (per lane)	O	N_L	RX Equalization in progress indicator. This signal can be left unconnected if not present on the PHY.
pipe_invalidrequest (per lane)	O	N_L	Indicates that the link evaluation feedback resulted in coefficients that were either illegal or rejected by the link partner.
pl_eq_txdeemph (per lane)	O	18*N_L	TX de-emphasis/TX coefficients to PHY Only for Gen3 or above speed PHY
pl_eq_fs (per lane)	O	6*N_L	FS value received from link partner
pl_eq_lf (per lane)	O	6*N_L	LF value received from link partner
pl_eq_rxpresethint (per lane)	O	3*N_L	RX Preset Hint value received from link partner. This signal can be left unconnected if not present on the PHY.
pl_eq_linkevaluationfeedbackfiguremerit (per lane)	I	8*N_L	Evaluation result in "figure of merit" format. Must be tied to 0's if this output is not available on the PHY. For MediaTek PHY: it is 1 bit signal in PHY and please replicate it to 8-bit signals and connect to MAC. For Standard PHY: it is 8-bit signal for each lane.
pl_eq_linkevaluationfeedbackdirectionchange (per lane)	I	6*N_L	Evaluation result in "direction change" format. Must be tied to 0's if this output is not available on the PHY. Note that bits [3:2] (corresponding to C0) are ignored.
pl_eq_txpresetcoefficients (per lane)	O	18*N_L	Indicate Coefficient value received from link partner's transmitter. Proprietary signal for MediaTek PHY. This signal can be left unconnected if not present on the PHY.

3.12.7.11.6 IO Related Signals

Signal Name	Type	Description	Ball Location
PCIe Port 0			
PCIEG3_LN0_RXP	AI	Lane 0 receive data differential pair	Y35
PCIEG3_LN0_RXN	AI		W35
PCIEG3_LN0_TXP	AO	Lane 0 transmit data differential pair	AA30
PCIEG3_LN0_TXN	AO		AA31
PCIEG3_LN1_RXP	AI	Lane 1 receive data differential pair	AC35
PCIEG3_LN1_RXN	AI		AB35
PCIEG3_LN1_TXP	AO	Lane 1 transmit data differential pair	AA37
PCIEG3_LN1_TXN	AO		AA36
PCIEG3_CLKP	AI	Reference clock differential pair	AA34
PCIEG3_CLKN	AI		AA33
PCIe Port 1 (Shared with USB Port 1)			
PCIE_LN0_RXP_P1	AI	Lane 0 receive data differential pair	V33
PCIE_LN0_RXN_P1	AI		V34
PCIE_LN0_TXP_P1	AO	Lane 0 transmit data differential pair	W31
PCIE_LN0_TXN_P1	AO		W32
PCIE_CKRXP_P1	AI	Reference clock differential pair	V36
PCIE_CKRXN_P1	AI		V37
PCIe Control Signals Port 0			
PERSTN	DO	Fundamental reset	AL28, AL24, AK16
CLKREQN	DIO	Clock request	AL29, AM24, AT16
WAKEN	DI	Link reactivation	AM29, AM23, AL16
PCIe Control Signals Port 1			
PERSTN_1	DO	Fundamental reset	H31, AL27, AN15
CLKREQN_1	DIO	Clock request	G34, AK27, AK15
WAKEN_1	DI	Link reactivation	J31, AM28, AU17

3.12.7.11.7 Interrupt Related Signals

Signal	I/O	Width	Description	Note
pcie_interrupt_out	O	1	PCIe main interrupt output. It is a level signal and is active high.	tl_clk domain

3.12.8 Keypad Scanner (Keypad)

3.12.8.1 Keypad Overview

The keypad module implements scanning algorithm for hardware-based key press decoding and reduces CPU overhead.

3.12.8.2 Keypad Features

The keypad module supports the following features:

- Two types of keyboards:
  - 3 × 3 single keys
  - 3 × 3 configurable double keys
  - Configurable key debounce time
- Double keypad supports 18-key matrix divided into 9 subgroups of 2 keys and a 20 Ω resistor
- Key detection block providing key press, key release and de-bounce mechanisms
- Interrupt event detection for key presses and key releases
- Detection of one or two keys pressed simultaneously with any combination

3.12.8.3 Keypad Block Diagram

The keypad module contains 2 submodules, KP0 and KP1. KP0 supports single keys, and KP1 supports double keys. Figure 3-244 and Figure 3-245 show the structural block diagrams of Keypad Top and KP1.

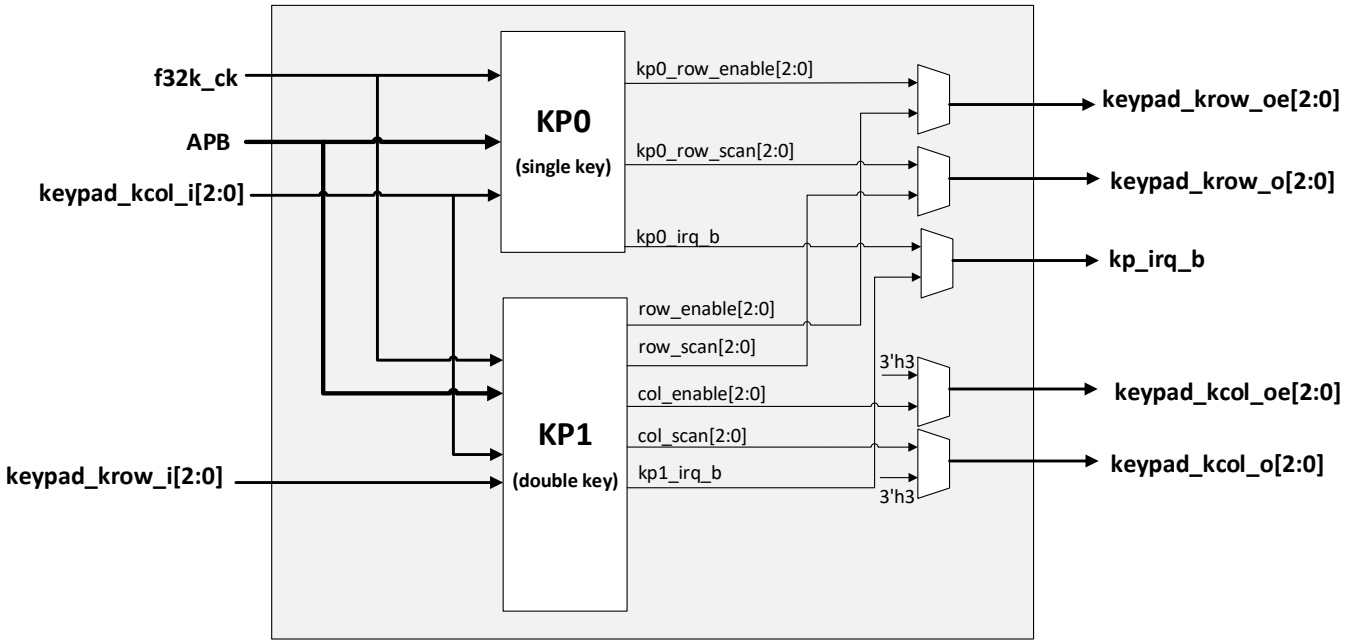


Figure 3-244 Block Diagram of Keypad Top

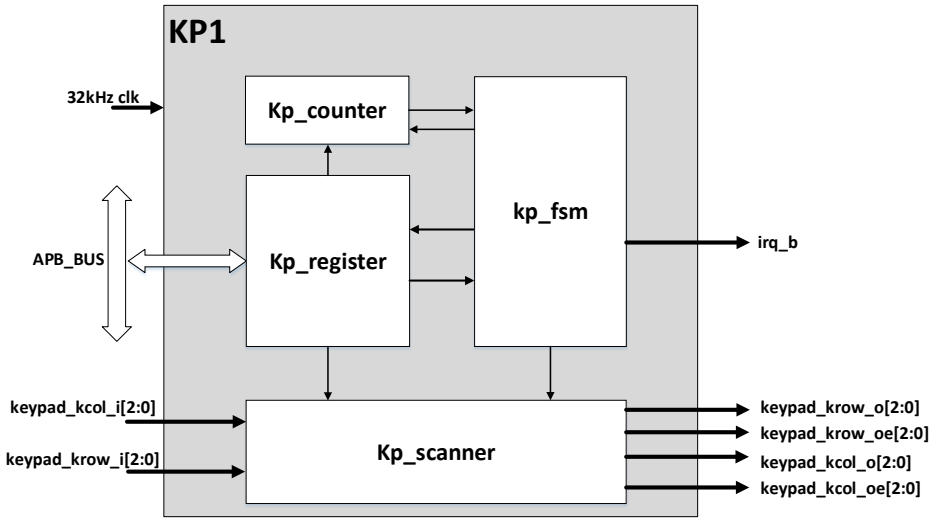


Figure 3-245 KP1 Block Diagram

KP1 comprises the following modules:

Table 3-181 KP1 Module Description

Module	Description
Kp_register	The register control module, housing the registers the APB can read and write. It is also responsible for sampling the present state of the keys.
Kp_scanner	Scans the keypad state.
Kp_FSM	Manages the working flow of the keypad and generates an IRQ signal.
Kp_counter	Counts the de-bounce time.

3.12.8.4 Keypad Signal Descriptions

Table 3-182 presents keypad signal descriptions.

Table 3-182 Keypad Signal Descriptions

Signal Name	Type	Description	Ball Location
KPCOL0	DIO	Keypad column 0	AU11
KPCOL1	DIO	Keypad column 1	AT11, AL10
KPCOL2	DIO	Keypad column 2	AT10
KPROW0	DIO	Keypad row 0	AN11
KPROW1	DIO	Keypad row 1	AN10
KPROW2	DIO	Keypad row 2	AM8

3.12.8.5 Keypad Function Description

The keypad module supports two types of keypads: 3 × 3 single keys and 3 × 3 configurable double keys. The keypad interface includes 3 columns and 3 rows (see Figure 3-244). The key detection block provides the key pressed, key released and de-bounce mechanisms.

Each time the key is pressed or released, i.e. something different in the 3 × 3 matrix, the key detection block senses the change and recognizes if a key has been pressed or released. Whenever the key status changes and becomes stable, a keypad IRQ is issued. Then, the MCU can read the key(s) pressed directly in the KP_MEM1, KP_MEM2 registers. The status register can only be changed by the key press detection FSM.

3.12.8.6 Keypad Theory of Operations

Key press detection depends on the HIGH or LOW level of the external keypad interface. If the keys are pressed at the same time, and a key shares the same column and the same row with other keys, the pressed key cannot be correctly decoded. For example, if there are three keys pressed: key1 = (x1, y1), key2 = (x2, y2), and key3 = (x1, y2), both key3 and key4 = (x2, y1) are detected, and therefore, they cannot be distinguished correctly. Hence, the keypad detects only one key or two keys pressed simultaneously in any combination. Pressing more than two keys simultaneously in a specific pattern may cause wrong information retrieval.

When the state of key matrix changes, there is a de-bounce time. Set suitable de-bounce time before enabling Keypad. If the time value is set too small, Keypad becomes overly sensitive and detects multiple unexpected key presses.

3.12.8.7 Keypad Programming Guide

Table 3-183 presents Keypad setting flow.

Table 3-183 Keypad Setting Flow

Step	Address	Register Name	MSB	LSB	Local Address	R/W	Default Value	Description
Set up keypad								
1	keypad base address + 0x020	KP_SEL	0	0	KP_SEL	RW	1'b0	Select the single keypad or the double keypad function
2	keypad base address + 0x018	KP_DEBOUNCE	13	0	DEBOUNCE	RW	14'h400	Set de-bounce time. De-bounce time = KP_DEBOUNCE/32 ms.
3	keypad base address + 0x024	KP_EN	0	0	KP_EN	RW	1'd0	Enable keypad scanner
Wait for keypad to issue an interrupt and read keypad MEM register.								
4	keypad base address + 0x000	KP_STA	0	0	STA	RO	1'b0	Read keypad status 0: No key pressed 1: Key pressed
5	keypad base address + 0x004	KP_MEM1	15	0	KEY15~KEY0	RO	16'hffff	Read keypad MEM register
6	keypad base address + 0x008	KP_MEM2	15	0	KEY31~KEY16	RO	16'hffff	Read keypad MEM register

3.12.8.8 Keypad Applications

The 3 × 3 double Keypad supports a 3 × 3 × 2 = 18-key matrix. The 18 keys are divided into 9 subgroups, and each group consists of 2 keys and a 20 Ω resistor.

Figure 3-246 represents 3 × 3 double Keypad matrix (18 keys) example configuration.

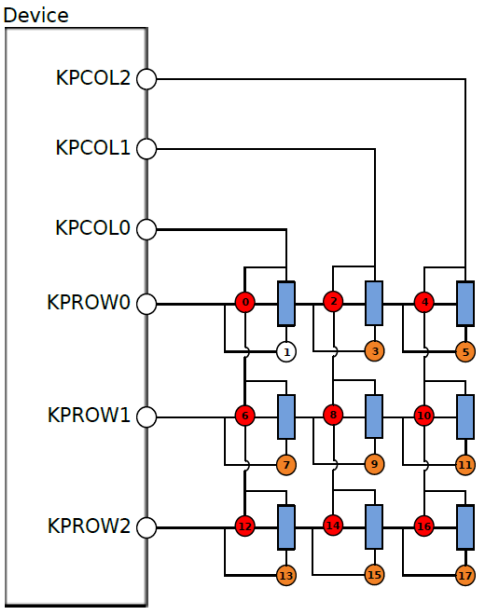


Figure 3-246 3 × 3 KeyPad Matrix (18 Keys)

3.12.9 General-purpose Input/Output (GPIO)

3.12.9.1 Overview

The General-purpose Input/Output (GPIO) peripheral provides dedicated pins that are configurable as either inputs or outputs. Each GPIO pin has the following key functions:

- Configurable direction: Input or output
- Control of the state driven on the output pin when GPIO is configured as an output
- Detection of the state of the pin when GPIO is configured as an input
- Configurable interrupt event generation

3.12.9.2 Features

3.12.9.2.1 Pad Pin List

The MT8395 offers the following types of pin control for GPIO. The corresponding I/O cells are described in the tables below.

I/O Cell	Table
General 1.8V I/O cell	Table 3-186
KP2K I/O cell	Table 3-187

I/O Cell	Table
KP200K I/O cell	Table 3-188
I3C I/O cell	Table 3-189
RST I/O cell	Table 3-190
MSDC3A18 I/O cell	Table 3-191
MSDC3A18OD33 I/O cell	Table 3-192
AGPIO General 1.8V I/O cell	Table 3-193

Table 3-184 GPIO Pin Control

Pin	Description
E pin	The pad can be used in either Transmit (TX) or Receive (RX) mode, which is configured by the E pin of the pad. • E = 1, TX mode, the TX path is from I to I/O. • E = 0, RX mode, the RX path is from I/O to O.
E2/E4/E8 pin	To configure the TX driving strength. See Table 3-185 for details.
IES pin	To control the RX input buffer. • IES = 1, the RX path is enabled • IES = 0, the RX path is disrupted, and O pin is 0
SMT pin	Schmitt trigger hysteresis control
PU/PD pin	To pull up or pull down the resistor.

Table 3-185 Driving Strength Control

{E8, E4, E2}	Driving Strength (mA)
3'b000	2
3'b001	4
3'b010	6
3'b011	8
3'b100	10
3'b101	12
3'b110	14
3'b111	16

3.12.9.2.2 I/O Cell

Table 3-186 General 1.8V I/O Cell

Pin Description	Type	Level	Description	Note
I	Input	0.75V	Input signal of driver	
IO	Input/ Output	1.8V	Data input/output. Output pin of driver (TX mode)/input pin of receiver (RX mode)	
E	Input	0.75V	To enable output. High asserted. E = 1, TX mode; E = 0, RX mode.	

Pin Description	Type	Level	Description	Note
E2, E4, E8	Input	0.75V	TX driving strength control	Register programmable 2G Type (E8, E4, E2): 2mA: [000] 4mA: [001] 6mA: [010] 8mA: [011] 10mA: [100] 12mA: [101] 14mA: [110] 16mA: [111]
O	Output	0.75V	Output signal of RX receiver	
IES	Input	0.75V	To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0 For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.	
SMT	Input	0.75V	To enable RX input buffer Schmitt trigger hysteresis control. High asserted. SMT= 1, Schmitt trigger enabled	Register programmable
PU	Input	0.75V	75K pull-up resistor control. High activation	Register programmable
PD	Input	0.75V	75K pull-down resistor control. High activation	Register programmable
RDSEL[1:0]	Input	0.75V	RX duty selection • RDSEL[0]: Level shifter duty high when asserted (high pulse width adjustment) • RDSEL[1]: Level shifter duty low when asserted (low pulse width adjustment)	Register programmable Default setting: 1.8V: RDSEL[1:0]=[00]
TDSEL[3:0]	Input	0.75V	TX duty selection • TDSEL[2][0]: Output level shifter duty high when asserted (high pulse width adjustment) • TDSEL[3][1]: Output level shifter duty low when asserted (low pulse width adjustment)	Register programmable Default setting: 1.8V: TDSEL[3:0]=[0000] 1.2V: TDSEL[3:0]=[0000] *Please set BIAS control pin, 1.8V: BSEL0=0, BDSEL[1:0]=[00] 1.2V: BSEL0=1, BDSEL[1:0]=[00] *Please set TDSEL[3:0]=[1111] @ if VCCK is lower than 0.4V.

Table 3-187 KP 2K I/O Cell

<table>
<tr><th>Pin Description</th><th>Type</th><th>Level</th><th>Description</th><th>Note</th></tr>
<tr><td>I</td><td>Input</td><td>0.75V</td><td>Input signal of driver</td><td></td></tr>
<tr><td>IO</td><td>Input/ Output</td><td>1.8V</td><td>Data input/output.
Output pin of driver (TX mode)/input pin of receiver (RX mode)</td><td></td></tr>
<tr><td>E</td><td>Input</td><td>0.75V</td><td>To enable output. High asserted.
E = 1, TX mode; E = 0, RX mode.</td><td></td></tr>
<tr><td>E2, E4, E8</td><td>Input</td><td>0.75V</td><td>TX driving strength control</td><td>Register programmable 2G Type
(E8, E4, E2):
<table><tr><td>2mA: [000]
4mA: [001]
6mA: [010]
8mA: [011]</td><td>10mA: [100]
12mA: [101]
14mA: [110]
16mA: [111]</td></tr></table></td></tr>
<tr><td>O</td><td>Output</td><td>0.75V</td><td>Output signal of RX receiver</td><td></td></tr>
<tr><td>IES</td><td>Input</td><td>0.75V</td><td>To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0
For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.</td><td></td></tr>
<tr><td>SMT</td><td>Input</td><td>0.75V</td><td>To enable RX input buffer Schmitt trigger hysteresis control. High asserted.
SMT= 1, Schmitt trigger enabled</td><td>Register programmable</td></tr>
<tr><td>PUPD</td><td>Input</td><td>0.75V</td><td rowspan="3"><table><tr><th>PUPD</th><th>R1</th><th>R0</th><th>R value</th></tr><tr><td>0</td><td>0</td><td>0</td><td>High Z</td></tr><tr><td>0</td><td>0</td><td>1</td><td>PU-75K</td></tr><tr><td>0</td><td>1</td><td>0</td><td>PU-2K</td></tr><tr><td>0</td><td>1</td><td>1</td><td>PU-75K/2K</td></tr><tr><td>1</td><td>0</td><td>0</td><td>High Z</td></tr><tr><td>1</td><td>0</td><td>1</td><td>PD-75K</td></tr><tr><td>1</td><td>1</td><td>0</td><td>PD-2K</td></tr><tr><td>1</td><td>1</td><td>1</td><td>PD-75K/2K</td></tr></table>75K/2K means parallel connection of resistance.</td><td>Register programmable</td></tr>
<tr><td>R0</td><td>Input</td><td>0.75V</td><td>Register programmable</td></tr>
<tr><td>R1</td><td>Input</td><td>0.75V</td><td>Register programmable</td></tr>
<tr><td>RDSEL[1:0]</td><td>Input</td><td>0.75V</td><td>RX duty selection
• RDSEL[0]: Level shifter duty high when asserted (high pulse width adjustment)
• RDSEL[1]: Level shifter duty low when asserted (low pulse width adjustment)</td><td>Register programmable
Default setting: 1.8V: RDSEL[1:0]=[00]</td></tr>
<tr><td>TDSEL[3:0]</td><td>Input</td><td>0.75V</td><td>TX duty selection</td><td>Register programmable
Default setting:</td></tr>
</table>

Pin Description	Type	Level	Description	Note
			• TDSEL[2][0]: Output level shifter duty high when asserted (high pulse width adjustment) • TDSEL[3][1]: Output level shifter duty low when asserted (low pulse width adjustment)	1.8V: TDSEL[3:0]=[0000] 1.2V: TDSEL[3:0]=[0000] *Please set BIAS control pin, 1.8V: BSEL0=0, BDSEL[1:0]=[00] 1.2V: BSEL0=1, BDSEL[1:0]=[00] *Please set TDSEL[3:0]=[1111] @ if VCCK is lower than 0.4V.

Table 3-188 KP 200K I/O Cell

Pin Description	Type	Level	Description	Note
I	Input	0.75V	Input signal of driver	
IO	Input/ Output	1.8V	Data input/output. Output pin of driver (TX mode)/input pin of receiver (RX mode)	
E	Input	0.75V	To enable output. High asserted. E = 1, TX mode; E = 0, RX mode.	
E2, E4, E8	Input	0.75V	TX driving strength control	Register programmable 2G Type (E8, E4, E2): 2mA: [000] 10mA: [100] 4mA: [001] 12mA: [101] 6mA: [010] 14mA: [110] 8mA: [011] 16mA: [111]
O	Output	0.75V	Output signal of RX receiver	
IES	Input	0.75V	To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0 For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.	
SMT	Input	0.75V	To enable RX input buffer Schmitt trigger hysteresis control. High asserted. SMT= 1, Schmitt trigger enabled	Register programmable
PUPD	Input	0.75V		Register programmable
R0	Input	0.75V		Register programmable
R1	Input	0.75V		Register programmable

PUPD	R1	R0	R value
0	0	0	High Z

Pin Description	Type	Level	Description	Note
			0 0 1 PU-75K 0 1 0 PU-200K 0 1 1 PU-75K/200K 1 0 0 High Z 1 0 1 PD-75K 1 1 0 PD-200K 1 1 1 PD-75K/200K 75K/200K means parallel connection of resistance.	
RDSEL[1:0]	Input	0.75V	RX duty selection • RDSEL[0]: Level shifter duty high when asserted (high pulse width adjustment) • RDSEL[1]: Level shifter duty low when asserted (low pulse width adjustment)	Register programmable Default setting: 1.8V: RDSEL[1:0]=[00]
TDSEL[3:0]	Input	0.75V	TX duty selection • TDSEL[2][0]: Output level shifter duty high when asserted (high pulse width adjustment) • TDSEL[3][1]: Output level shifter duty low when asserted (low pulse width adjustment)	Register programmable Default setting: 1.8V: TDSEL[3:0]=[0000] 1.2V: TDSEL[3:0]=[0000] *Please set BIAS control pin, 1.8V: BSEL0=0, BDSEL[1:0]=[00] 1.2V: BSEL0=1, BDSEL[1:0]=[00] *Please set TDSEL[3:0]=[1111] @ if VCCK is lower than 0.4V.

Table 3-189 I3C I/O Cell

Pin Description	Type	Level	Description	Note
I	Input	0.75V	Input signal of driver	
IO	Input/ Output	1.8V	Data input/output. Output pin of driver (TX mode)/input pin of receiver (RX mode)	
E	Input	0.75V	To enable output. High asserted. E = 1, TX mode; E = 0, RX mode.	
E2, E4, E8	Input	0.75V	TX driving strength control	Register programmable 2G Type (E8, E4, E2): 2mA: [000] 10mA: [100] 4mA: [001] 12mA: [101] 6mA: [010] 14mA: [110] 8mA: [011] 16mA: [111]

Pin Description	Type	Level	Description	Note
EH	Input	0.75V	To enable the I2C mode.	Register programmable
EH1, EH2	Input	0.75V	TX driving strength control for the I2C mode	Register programmable
O	Output	0.75V	Output signal of the RX	
IES	Input	0.75V	To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0 For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.	
SMT	Input	0.75V	To enable RX input buffer Schmitt trigger hysteresis control. High asserted. SMT= 1, Schmitt trigger enabled	Register programmable
PU	Input	0.75V	75K pull-up resistor control. High activation	Register programmable
PD	Input	0.75V	75K pull-down resistor control. High activation	Register programmable
RSEL	Input	0.75V	Pull-up/down resistance selection. • RSEL=0, R=75K (GPIO mode) • RSEL=1, RSEL1=0, R=5K (SoundWire IO mode)	
RSEL1	Input	0.75V	Additional strong pull resistor control for the I2C pull-up resistor	
RSEL2	Input	0.75V	Additional strong pull resistor control for the I2C pull-up resistor.	
RDSEL[1:0]	Input	0.75V	RX duty selection • RDSEL[0]: Level shifter duty high when asserted (high pulse width adjustment) • RDSEL[1]: Level shifter duty low when asserted (low pulse width adjustment)	RDSEL[1:0] = [00] Other settings are not allowed.
TDSEL[3:0]	Input	0.75V	TX duty selection • TDSEL[2][0]: Output level shifter duty high when asserted (high pulse width adjustment) • TDSEL[3][1]: Output level shifter duty low when asserted (low pulse width adjustment)	TDSEL[3:0] = [0000] Other settings are not allowed.

Table 3-190 RST I/O Cell

Pin Description	Type	Level	Description	Note
I	Input	0.75V	Input signal of driver	
O	Output	0.75V	Output signal of the RX	
IES	Input	0.75V	To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0 For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.	
IES18	Input	0.75V	O18_ANALOG_DOUNT_TOUCH control • IES18 = 1, O18_ANALOG_DOUNT_TOUCH = IO • IES18 = 0, O18_ANALOG_DOUNT_TOUCH = 0	Register programmable
SMT	Input	0.75V	To enable RX input buffer Schmitt trigger hysteresis control. High asserted.	Register programmable

Pin Description	Type	Level	Description	Note
			SMT= 1, Schmitt trigger enabled	
PU	Input	0.75V	75K pull-up resistor control. High activation	Register programmable
PD	Input	0.75V	75K pull-down resistor control. High activation	Register programmable
RDSEL[1:0]	Input	0.75V	RX duty selection • RDSEL[0]: Level shifter duty high when asserted (high pulse width adjustment) • RDSEL[1]: Level shifter duty low when asserted (low pulse width adjustment)	RDSEL[1:0] = [00] Other settings are not allowed.
O18_ANALOG_DOUNT_TOUCH	Output	1.8V	Analog IO. Will be connected to analog circuit (for system reset)	Register programmable
AIO_ANALOG_DONT_TOUCH	Output	1.8V/1.2V	Internal pin to core side. Connected to I by resistor	

Table 3-191 MSDC3A18 I/O Cell

Pin Description	Type	Level	Description	Note
I	Input	0.75V	Input signal of driver	
IO	Input/ Output	1.8V	Data input/output. Output pin of driver (TX mode)/input pin of receiver (RX mode)	
E	Input	0.75V	To enable output. High asserted. E = 1, TX mode; E = 0, RX mode.	
E2, E4, E8	Input	0.75V	TX driving strength control	Register programmable 2G Type (E8, E4, E2): 2mA: [000] 4mA: [001] 6mA: [010] 8mA: [011] 10mA: [100] 12mA: [101] 14mA: [110] 16mA: [111]
O	Output	0.75V	Output signal of RX receiver	
IES	Input	0.75V	To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0 For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.	
SMT	Input	0.75V	To enable RX input buffer Schmitt trigger hysteresis control. High asserted. SMT= 1, Schmitt trigger enabled	Register programmable
R0	Input	0.75V	PU/PD=0, R1=0, R0=0, R value=High Z	Register programmable
R1	Input	0.75V		Register programmable Register programmable

Pin Description	Type	Level	Description	Note
PUPD	Input	0.75V	0 0 1 PU-10K 0 1 0 PU-50K 0 1 1 PU-10K/50K 1 0 0 High Z 1 0 1 PD-10K 1 1 0 PD-50K 1 1 1 PD-10K/50K 10K/50K means parallel connection of resistance.	Register programmable
RDSEL[5:0]	Input	0.75V	RX duty select • RDSEL[1:0]: Input buffer duty high when asserted (high pulse width adjustment) • RDSEL[3:2]: Input buffer duty low when asserted(low pulse width adjustment) • RDSEL[5:4]: Level shifter duty high when asserted (high pulse width adjustment)	Register programmable Default setting: RDSEL[5:0]=000000
TDSEL[3:0]	Input	0.75V	TX duty select • TDSEL[1:0]: Output level shifter duty high when asserted (high pulse width adjustment) • TDSEL[3:2]: Output level shifter duty low when asserted(low pulse width adjustment)	Register programmable Default setting: TDSEL[3:0]=0000 TDSEL[3:0]=1111 if operating in low speed mode(VCCK= 0.45v +-10%)

Table 3-192 MSDC3A18OD33 I/O Cell

Pin Description	Type	Level	Description	Note
VCC3IO	Power	3V	3V Power Port. For IO output buffer	
VCC18I	Power	1.8V	1.8V Power Port. For IO pre-driver and RX path	
VCCK	Power	0.75V	0.75V Power Port. For IO control logic	
GND3IO	Ground	0V	Ground Port	
GNDK	Ground	0V	Ground Port	
I	Input	0.75V	Input Signal of Driver	
IO	Input/ Output	1.8V/3.0V	Data Input/Output. Output pin of Driver (TX mode)/Input Pin of Receiver (RX mode)	
E	Input	0.75V	Output Enalbe. High asserted. E=1, TX mode. E=0, RX mode.	
SR	Input	0.75V	Output Slew Rate Control. High asserted. SR=1, slower slew. SR=0, no slew rate controlled.	
E2, E4, E8	Input	0.75V	TX driving strength control	Register programmable 2G Type (E8, E4, E2):

<table>
<tr><th>Pin Description</th><th>Type</th><th>Level</th><th>Description</th><th colspan="2">Note</th></tr>
<tr><td></td><td></td><td></td><td></td><td>2mA: [000]
4mA: [001]
6mA: [010]
8mA: [011]</td><td>10mA: [100]
12mA: [101]
14mA: [110]
16mA: [111]</td></tr>
<tr><td>O</td><td>Output</td><td>0.75V</td><td>Output Signal of RX receiver</td><td colspan="2"></td></tr>
<tr><td>IES</td><td>Input</td><td>0.75V</td><td>RX input buffer enable. High asserted. Datapath: from IO to O. IES=0, O=0</td><td colspan="2">Power down mode: IES=0 is a must
Quiescent mode: IES=0 is suggested</td></tr>
<tr><td>SMT</td><td>Input</td><td>0.75V</td><td>RX input buffer Schmitt trigger hysteresis control enable. High asserted. SMT=1, Schmitt trigger enable</td><td colspan="2"></td></tr>
<tr><td>PUPD/R0/R1</td><td>Input</td><td>0.75V</td><td>PUPD/R0/R1 control pin of PU/PD. See the attached truth table below
PU/PD resistence SPEC
<table>
<tr><th>E</th><th>PUPD</th><th>R1</th><th>R0</th><th>R Value of 1.8V/3.3V IO Power</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>High-Z</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PU10Kohm</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>PU50Kohm</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>PU 10Kohm//50Kohm</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>High-Z</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>PD10Kohm</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>PD50Kohm</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>PD 10Kohm//50Kohm</td></tr>
<tr><td>1</td><td>X</td><td>X</td><td>X</td><td>High-Z</td></tr>
</table></td><td colspan="2">Register programmable</td></tr>
<tr><td>TDSEL[3:0]</td><td>Input</td><td>0.75V</td><td>• TDSEL0, TDSEL2: Output level shifter duty high when asserted (high pulse width adjustment)
• TDSEL1, TDSEL3: Output level shifter duty low when asserted (low pulse width adjustment)</td><td colspan="2">Register programmable</td></tr>
<tr><td>RDSEL[5:0]</td><td>Input</td><td>0.75V</td><td>• RDSEL0, RDSEL1, RDSEL4: input level shifter duty high when asserted (high pulse width adjustment)
• TDSEL2, TDSEL3, RDSEL5: input level shifter duty low when asserted (low pulse width adjustment)</td><td colspan="2">Register programmable</td></tr>
</table>

Table 3-193 AGPIO General 1.8V I/O Cell

Pin Description	Type	Level	Description	Notes
I	Input	0.75V	Input signal of driver	
IO	Input/ Output	1.8V	Data input/output. Output pin of driver (TX mode)/input pin of receiver (RX mode)	
E	Input	0.75V	To enable output. High asserted.	

Pin Description	Type	Level	Description	Notes
			E = 1, TX mode; E = 0, RX mode.	
E2, E4, E8	Input	0.75V	TX driving strength control	Register programmable 2G Type (E8, E4, E2): 2mA: [000] 4mA: [001] 6mA: [010] 8mA: [011] 10mA: [100] 12mA: [101] 14mA: [110] 16mA: [111]
O	Output	0.75V	Output signal of RX receiver	
IES	Input	0.75V	To enable RX input buffer. High asserted data path: from IO to O. IES = 0, O = 0 For VIO turn-off applications, IES = 0 prevents leakage in VDDIO power domain.	
SMT	Input	0.75V	To enable RX input buffer Schmitt trigger hysteresis control. High asserted. SMT= 1, Schmitt trigger enabled	Register programmable
PU	Input	0.75V	75K pull-up resistor control. High activation	Register programmable
PD	Input	0.75V	75K pull-down resistor control. High activation	Register programmable
RDSEL[1:0]	Input	0.75V	RX duty selection • RDSEL[0]: Level shifter duty high when asserted (high pulse width adjustment) • RDSEL[1]: Level shifter duty low when asserted (low pulse width adjustment)	Register programmable Default setting: 1.8V: RDSEL[1:0]=[00]
TDSEL[3:0]	Input	0.75V	TX duty selection • TDSEL[2][0]: Output level shifter duty high when asserted (high pulse width adjustment) • TDSEL[3][1]: Output level shifter duty low when asserted (low pulse width adjustment)	Register programmable Default setting: 1.8V: TDSEL[3:0]=[0000] 1.2V: TDSEL[3:0]=[0000] *Please set BIAS control pin, 1.8V: BSEL0=0, BDSEL[1:0]=[00] 1.2V: BSEL0=1, BDSEL[1:0]=[00] *Please set TDSEL[3:0]=[1111] @ if VCCK is lower than 0.4V.
AIO_ANALOG_DONT_TOUCH	Input/Output	1.8V	Analog IO. Will be connected to analog circuit	

Pin Description	Type	Level	Description	Notes
G	Input	0.75V	Analog/digital IO function selection port • G=0 => analog IO mode • G=1 => digital IO mode	When G=0, analog IO mode When G=1, digital IO mode

3.12.9.3 Signal Multiplexing

Each GPIO pin of the MT8395 may have up to 8 auxiliary functions (Aux Func.0 to Aux Func.7) that can be either output or input. The following sections explain the output and input signal multiplexing cases.

- Setting the GPIO_DIR register to **1** configures the GPIO as the **output** mode.
- Setting the GPIO_DIR register to **0** configures the GPIO as the **input** mode.

3.12.9.3.1 Output Signal Multiplexing

- When the *GPIO_MODE* register is set to be 0, the GPIO function (i.e., Aux Func.0) is selected. Additional registers, such as *GPIO_DOUT*, are also programmable to control the output state (e.g., output low or output high).
- When the *GPIO_MODE* register is set to any other values, the corresponding pin-mux function, for example, Aux Func.1 or Aux Func.2, is selected.

3.12.9.3.2 Input Signal Multiplexing

- When the *GPIO_MODE* register is set to 0, the GPIO function (i.e., Aux Func.0) is selected. Additional registers such as PU and PD are also programmable for other required usages, such as input high-Z, input pull down, or input pull high. Note that the PU and PD settings for each pin are separated into different control register domains.
- When the *GPIO_MODE* register is set to other values, one of the other pin mux functions such as Aux Func.1 or Aux Func.2, is selected.

3.12.9.4 Bitwise Command Register

To explain the bitwise command registers, take the following registers related to the same GPIO pads input/out direction as an example.

Register	Base Address	Description
GPIO_DIR0	(GPIO Base address+0x0000)[31:0]	Requires a “read-modify-write” operation to set the I/O direction for the corresponding GPIO pads.
GPIO_DIR0_SET	(GPIO Base address+0x0004)[31:0]	Bitwise functions • Bits written with "1" in *GPIO_DIR0_SET* set the corresponding GPIO pads as output pins. • Bits written with "1" in *GPIO_DIR0_CLR* set the corresponding GPIO pads as input pins. • Bits written with “0” have no effect, i.e., no read-modify-write operation is needed for the bitwise registers.
GPIO_DIR0_CLR	(GPIO Base address+0x0008)[31:0]	

Register	Base Address	Description
GPIO_DOUT GPIO_DOUT_SET GPIO_DOUT_CLR	-	Set the output value of 0/1 if the corresponding pads are configured as output pins.

See below for the example of writing the output value through the following registers.

Register	Base Address	Description
GPIO_DOUT0	(GPIO Base address+0x0100)[31:0]	Requires a “read-modify-write” operation to set the output value for the corresponding GPIO pads.
GPIO_DOUT0_SET	(GPIO Base address+0x0104)[31:0]	Bitwise functions • Writing "1" to the bits in *GPIO_DOUT0_SET* sets the corresponding GPIO pads output value to 1. • Writing “1” to bits in *GPIO_DOUT0_CLR* sets the corresponding GPIO pads output value to 0. • Bits written with “0” have no effect, i.e., no read-modify-write operation is needed for the bitwise registers.
GPIO_DOUT0_CLR	(GPIO Base address+0x0108)[31:0]	

3.12.9.5 Block Diagram

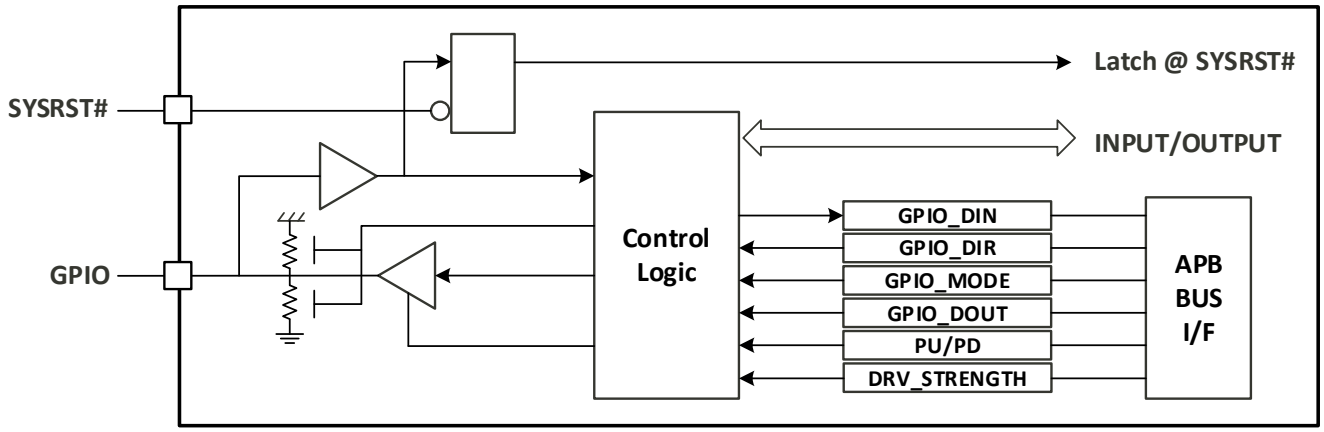


Figure 3-247 GPIO Block Diagram

3.12.9.6 GPIO Signal Descriptions

Table 3-194 presents GPIO signal descriptions.

Table 3-194 GPIO Signal Descriptions

Signal Name	Type	Description	Ball Location
GPIO0	DIO	General-purpose input and output	H31
GPIO1	DIO	General-purpose input and output	G34
GPIO2	DIO	General-purpose input and output	J31
GPIO3	DIO	General-purpose input and output	G33
GPIO4	DIO	General-purpose input and output	G35

Signal Name	Type	Description	Ball Location
GPIO5	DIO	General-purpose input and output	H33
GPIO6	DIO	General-purpose input and output	H32
GPIO7	DIO	General-purpose input and output	H34
GPIO8	DIO	General-purpose input and output	D36
GPIO9	DIO	General-purpose input and output	D37
GPIO10	DIO	General-purpose input and output	E36
GPIO11	DIO	General-purpose input and output	E35
GPIO12	DIO	General-purpose input and output	E37
GPIO13	DIO	General-purpose input and output	F37
GPIO14	DIO	General-purpose input and output	F35
GPIO15	DIO	General-purpose input and output	F34
GPIO16	DIO	General-purpose input and output	F36
GPIO17	DIO	General-purpose input and output	G36
GPIO18	DIO	General-purpose input and output	AP25
GPIO19	DIO	General-purpose input and output	AM29
GPIO20	DIO	General-purpose input and output	AL28
GPIO21	DIO	General-purpose input and output	AL29
GPIO22	DIO	General-purpose input and output	AL27
GPIO23	DIO	General-purpose input and output	AK27
GPIO24	DIO	General-purpose input and output	AM28
GPIO25	DIO	General-purpose input and output	AL26
GPIO26	DIO	General-purpose input and output	AM26
GPIO27	DIO	General-purpose input and output	AN25
GPIO28	DIO	General-purpose input and output	AN24
GPIO29	DIO	General-purpose input and output	AM25
GPIO30	DIO	General-purpose input and output	AP24
GPIO31	DIO	General-purpose input and output	AL24
GPIO32	DIO	General-purpose input and output	AM24
GPIO33	DIO	General-purpose input and output	AM23
GPIO34	DIO	General-purpose input and output	AL23
GPIO35	DIO	General-purpose input and output	AL25
GPIO36	DIO	General-purpose input and output	AR21
GPIO37	DIO	General-purpose input and output	AM22
GPIO38	DIO	General-purpose input and output	AM21
GPIO39	DIO	General-purpose input and output	AL21
GPIO40	DIO	General-purpose input and output	AU21
GPIO41	DIO	General-purpose input and output	AT21
GPIO42	DIO	General-purpose input and output	AP21
GPIO43	DIO	General-purpose input and output	AN21
GPIO44	DIO	General-purpose input and output	AN20
GPIO45	DIO	General-purpose input and output	AM20
GPIO46	DIO	General-purpose input and output	AL20
GPIO47	DIO	General-purpose input and output	AK20
GPIO48	DIO	General-purpose input and output	AR20

Signal Name	Type	Description	Ball Location
GPIO49	DIO	General-purpose input and output	AP20
GPIO50	DIO	General-purpose input and output	AN19
GPIO51	DIO	General-purpose input and output	AK19
GPIO52	DIO	General-purpose input and output	AT19
GPIO53	DIO	General-purpose input and output	AR18
GPIO54	DIO	General-purpose input and output	AP18
GPIO55	DIO	General-purpose input and output	AN18
GPIO56	DIO	General-purpose input and output	AM18
GPIO57	DIO	General-purpose input and output	AL18
GPIO58	DIO	General-purpose input and output	AK18
GPIO59	DIO	General-purpose input and output	AU20
GPIO60	DIO	General-purpose input and output	AT20
GPIO61	DIO	General-purpose input and output	AP17
GPIO62	DIO	General-purpose input and output	AL17
GPIO63	DIO	General-purpose input and output	AU18
GPIO64	DIO	General-purpose input and output	AT18
GPIO65	DO	General-purpose input and output	AR16
GPIO66	DIO	General-purpose input and output	AP16
GPIO67	DI	General-purpose input and output	AN16
GPIO68	DIO	General-purpose input and output	AM16
GPIO69	DIO	General-purpose input and output	AL16
GPIO70	DIO	General-purpose input and output	AK16
GPIO71	DIO	General-purpose input and output	AT16
GPIO72	DIO	General-purpose input and output	AN15
GPIO73	DIO	General-purpose input and output	AK15
GPIO74	DIO	General-purpose input and output	AU17
GPIO75	DIO	General-purpose input and output	AT17
GPIO76	DIO	General-purpose input and output	AT15
GPIO77	DIO	General-purpose input and output	AN14
GPIO78	DIO	General-purpose input and output	AM14
GPIO79	DIO	General-purpose input and output	AL14
GPIO80	DIO	General-purpose input and output	AK14
GPIO81	DIO	General-purpose input and output	AU15
GPIO82	DIO	General-purpose input and output	AR14
GPIO83	DIO	General-purpose input and output	AK11
GPIO84	DIO	General-purpose input and output	AK10
GPIO85	DIO	General-purpose input and output	AP14
GPIO86	DIO	General-purpose input and output	AR12
GPIO87	DIO	General-purpose input and output	AL12
GPIO88	DIO	General-purpose input and output	AK12
GPIO89	DIO	General-purpose input and output	AP12
GPIO90	DIO	General-purpose input and output	AL13
GPIO91	DIO	General-purpose input and output	AP13
GPIO92	DIO	General-purpose input and output	AN12

Signal Name	Type	Description	Ball Location
GPIO93	DIO	General-purpose input and output	AU14
GPIO94	DIO	General-purpose input and output	AT13
GPIO95	DIO	General-purpose input and output	AT12
GPIO96	DIO	General-purpose input and output	AT14
GPIO97	DIO	General-purpose input and output	AP9
GPIO98	DIO	General-purpose input and output	AR10
GPIO99	DIO	General-purpose input and output	AP10
GPIO100	DIO	General-purpose input and output	AM8
GPIO101	DIO	General-purpose input and output	AT10
GPIO102	DIO	General-purpose input and output	AR8
GPIO103	DIO	General-purpose input and output	AP8
GPIO104	DIO	General-purpose input and output	AN11
GPIO105	DIO	General-purpose input and output	AN10
GPIO106	DIO	General-purpose input and output	AU11
GPIO107	DIO	General-purpose input and output	AT11
GPIO108	DIO	General-purpose input and output	AL10
GPIO109	DIO	General-purpose input and output	AT9
GPIO110	DIO	General-purpose input and output	AN8
GPIO111	DIO	General-purpose input and output	AU9
GPIO112	DIO	General-purpose input and output	AT6
GPIO113	DIO	General-purpose input and output	AT7
GPIO114	DIO	General-purpose input and output	AT8
GPIO115	DIO	General-purpose input and output	AU8
GPIO116	DIO	General-purpose input and output	E32
GPIO117	DIO	General-purpose input and output	E33
GPIO118	DIO	General-purpose input and output	E34
GPIO119	DIO	General-purpose input and output	D35
GPIO120	DIO	General-purpose input and output	B35
GPIO121	DIO	General-purpose input and output	C36
GPIO122	DIO	General-purpose input and output	B36
GPIO123	DIO	General-purpose input and output	C35
GPIO124	DIO	General-purpose input and output	D32
GPIO125	DIO	General-purpose input and output	D33
GPIO126	DIO	General-purpose input and output	D34
GPIO127	DIO	General-purpose input and output	A35
GPIO128	DIO	General-purpose input and output	AL9
GPIO129	DIO	General-purpose input and output	AM10
GPIO130	DIO	General-purpose input and output	AL7
GPIO131	DIO	General-purpose input and output	AL8
GPIO132	DIO	General-purpose input and output	F31
GPIO133	DIO	General-purpose input and output	F32
GPIO134	DIO	General-purpose input and output	G32
GPIO135	DIO	General-purpose input and output	G31
GPIO136	DIO	General-purpose input and output	AT22

Signal Name	Type	Description	Ball Location
GPIO137	DIO	General-purpose input and output	AU23
GPIO138	DIO	General-purpose input and output	AT23
GPIO139	DIO	General-purpose input and output	AR23
GPIO140	DIO	General-purpose input and output	AN23
GPIO141	DIO	General-purpose input and output	AP22
GPIO142	DIO	General-purpose input and output	AN22
GPIO143	DIO	General-purpose input and output	AP23

3.12.9.7 Function Description

The I/O pins of the MT8395 can be programmed for multiple purposes, such as GPIO, NAND and SPI, by setting the GPIO_MODE register for different functions. Note that all functions must comply with the priority rule.

- When there are more than one I/O set as the same **output** function, all of the selected I/Os are able to output specific signals.
- When there are more than one I/O set as the same **input** (or bi-directional) function, only the I/O with the largest GPIO index works functionally.

3.12.9.7.1 Main Functions

Table 3-195 Main Functions of GPIOs

Main Function	Description	Notes
GPIO_MODE	GPIO mode selection	• 000 as GPIO mode • 001 to 111 as Aux Func.
GPIO_DIR	GPIO direction	• 0 as input • 1 as output
GPIO_DOUT	GPIO output.	-
GPIO_DIN	GPIO input.	-
PU/PD	Pull up/down resistor control. High activation	• PU = 1, PD = 0 as pull-up • PU = 0, PD = 1 as pull-down • PU = 0, PD = 0 as no-pull
DRV_STRENGTH	Driving strength of the GPIO, which can be configured up to 16 mA.	• 2 mA: [000] • 4 mA: [001] • 6 mA: [010] • 8 mA: [011] • 10 mA: [100] • 12 mA: [101] • 14 mA: [110] • 16 mA: [111]
INPUT/OUTPUT	I/O to the Aux Func IP.	-
Latch	To latch the GPIO signal during startup. Used for strapping (for more details, refer to Section 3.12.9.7.2).	-

3.12.9.7.2 Strapping Functions

Table 3-196 lists the strapping pins. The strapping pin state is latched when the system "resetb" changes from low to high. The strapping pin state decides which stage mode the system enters, or where the BROM boots from. When the strapping stage is completed, the strapping pin state can be changed.

To exit the strapping mode, configure the register *TPBANK* in the GPIO. Refer to Table 3-197 for more details.

Table 3-196 Strapping Configurations

Pin Name	Description	TRAP_FORCE_EN	TRAP_FORCE_VALUE
[0] PAD_AUD_DAT_MOSI0	trap_out_io	Bit0	Bit7
[1] PAD_PWRAP_SPI_CSN		Bit1	Bit8
[0] PAD_AUD_CLK_MOSI	Boot selection mode	Bit2	Bit9
[1] PAD_AUD_SYNC_MOSI		Bit3	Bit10
PAD_AUD_DAT_MOSI1	emmc_bit_speed_mode	Bit4	Bit11
PAD_CMMCLK0	spi_nor_address_mode	Bit5	Bit12
PAD_SCP_VREQ_VAO	boot_room_boot_partition_sequence	Bit6	Bit13

Table 3-197 Strapping Configuration Register

Address and Bit	Name	Description
0x100056F0[6:0]	TRAP_FORCE_EN(JTAG)	To enable the strapping configuration with TRAP_FORCE_VALUE(JTAG) Active high
0x100056F0[13:7]	TRAP_FORCE_VALUE(JTAG)	To force the value to JTAG strapping.

3.12.9.7.3 Secure Function

All GPIO control registers have a secure function.

- Enable the secure function by the *sec_en* register.
- The secure function is disabled by default.
- If the *sec_en* register bit = 1, the GPIO register follows the register map security definition.

3.12.9.8 Theory of Operations

3.12.9.8.1 Select GPIO Mode

To enable certain function on certain pads, change the GPIO mode of the pad. When the *GPIO_MODE* register is set to 0, the GPIO function (i.e., Aux Func.0) is selected. When the *GPIO_MODE* register is set to any other values, the corresponding pin-mux function, for example, Aux Func.1 or Aux Func.2, is selected.

3.12.9.8.2 Configure Pad Output Value and Observe Pad Input

When a certain pad's GPIO mode is configured as 0, you can configure the pad output value and observe the input value. When the pad GPIO mode is not 0, you cannot configure the pad output value by GPIO setting. You can observe the pad input value when the pad holds on stable.

1. If you want to configure the pad output value, follow the steps below.
   Step 1: Set GPIO_MODE as 0
   Step 2: Set GPIO_DIR as 1
   Step 3: Setting GPIO_DOUT as 1 means output high, while setting GPIO_DOUT as 0 means output low
2. If you want to observe the pad input value, follow the steps below.
   Step 1: Set GPIO_MODE as 0
   Step 2: Set GPIO_DIR as 0
   Step 3: Setting GPIO_DIN as 1 means input high, while setting GPIO_DIN as 0 means input low

3.12.9.8.3 Configure Pins of Pads

The pins of pads can be configured when needed, but the default value should be sufficient for most scenarios.

3.12.9.9 Programming Guide

3.12.9.9.1 GPIO Mode as Input

Table 3-198 GPIO Mode Input Programming Outline

Step	Sequence	Register Name	Register Value	Address
1	Set the AUX function to GPIO Mode0.	GPIO_MODE	0	GPIO Base + GPIO_MODE
2	Set the GPIO DIR register to be configured as the input.	GPIO_DIR	0	GPIO Base + GPIO_DIR

3.12.9.9.2 GPIO Mode as Output

Table 3-199 GPIO Mode Output Programming Outline

Step	Sequence	Register Name	Register Value	Address
1	Set the AUX function to GPIO Mode0.	GPIO_MODE	0	GPIO Base + GPIO_MODE
2	Set the GPIO DIR register to be configured as the output.	GPIO_DIR	1	GPIO Base + GPIO_DIR
3	Set GPIO DOUT value	GPIO_DOUT	• Set 1 as output high • Set 0 as output low	GPIO Base + GPIO_DOUT

3.12.9.10 Register Definition

Refer to *"MT8395 Register Map"* for detailed register descriptions.

3.12.9.11 Electrical Characteristics

Table 3-200 Electrical Characteristics and Operating Conditions of 1.8V Applications

Parameter	Description	Min	Typ	Max	Unit
VCC18O	Supply voltage of IO power	1.62	1.8	1.98	V
Freq(1)	Maximum operating frequency	-	-	55	MHz
VCCK	Supply voltage of core power	0.495	0.75	0.825	V

Parameter	Description	Min	Typ	Max	Unit
Input					
V_{IH}	High-level input voltage	0.65 * VCC18O	-	VCC18O + 0.3	V
V_{IL}	Low-level input voltage	-0.3	-	0.35 * VCC18O	V
R_{pu}	Input pull-up resistance	40	75	190	KΩ
R_{pd}	Input pull-down resistance	40	75	190	KΩ
Output					
$V_{OH\ (DC)}$	High-level output voltage	0.75 * VCC18O	-		V
$V_{OL\ (DC)}$	Low-level output voltage	-	-	0.25*VCC18O	V
Leakage					
I_{IN}	Input leakage current (Any input 0V < VIN < VCC18O)	-10	-	10	µA
IVCC18Oz	VCC18O standby leakage @ I/O hi-z and RX off	-10	-	10	µA
IVCCKz	VCCK standby leakage @ I/O hi-z and RX off	-10	-	10	µA

(1) I3C mode maximum operating frequency = 3.4 MHz
MSDC I/O maximum operating frequency = 208 MHz
SPMI I/O maximum operating frequency = 120 MHz

3.12.10 Pulse Width Modulation (PWM)

3.12.10.1 Overview

The Pulse Width Modulation (PWM) is a technique designed to generate pulse sequences with programmable frequency and duration for various applications, including but not limited to Liquid-Crystal Display (LCD) backlight and charging.

Prior to enabling the PWM, the pulse sequences must be prepared in the registers. The PWM then reads the stored pulse sequences to generate the expected waveform to meet specific requirements of the given applications.

3.12.10.2 Features

Different PWM modes as listed below are available.

Table 3-201 Pulse Width Modulation (PWM) Modes

Mode	Description
Normal Mode	Generates pulse sequences with programmable frequency and duration.
FIFO Mode	Generates pulse sequences for the data that is programmable by the CPU.
Memory Mode and Random Mode	Generate pulse sequence for the data sent from SRAM.

3.12.10.3 Block Diagram

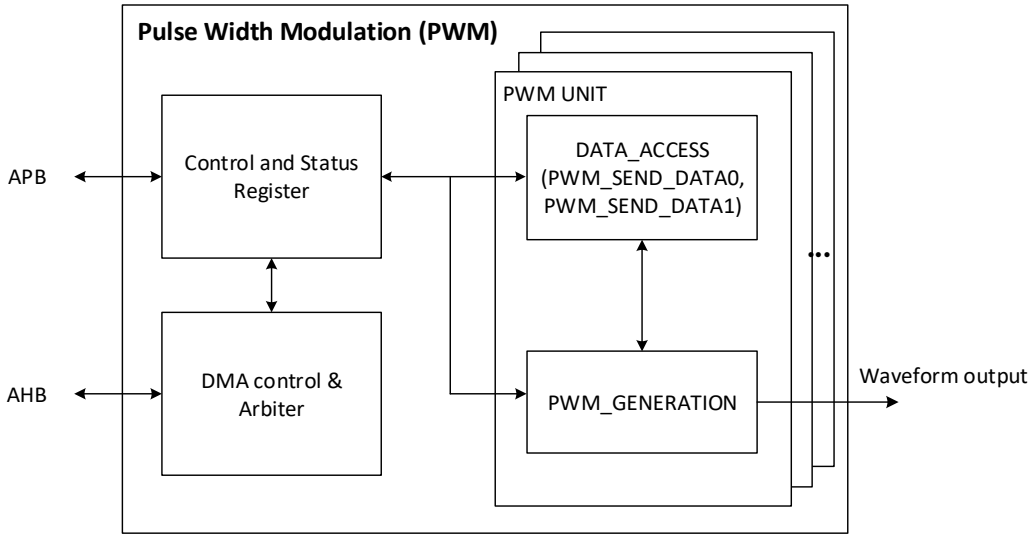


Figure 3-248 Block Diagram of PWM

Figure 3-248 illustrates the PWM block diagram, which consists of several functional components:

- **Control and status registers**: Receive commands from the system.
- **DMA control and arbitrator**: Facilitates communication between the system RAM and the PWM when the normal PWM is set to the memory mode or random mode.
- **DATA_ACCESS**: Manages the data flow from the AHB/APB.
- **PWM_GENERATION**: Controls the transmission of the PWM waveform.

3.12.10.4 Function Description

3.12.10.4.1 Waveform Generation

The PWM outputs waveforms through the PAD pins. The theoretical methodology for generating PWM waveforms differs across modes, as outlined in Section 3.12.10.5. The PWM generates waveforms by dividing frequencies and data sequences in a specific manner as described in the following sections.

3.12.10.4.1.1Frequency Division Generation

The PWM generates a waveform by dividing the frequency in the normal mode.

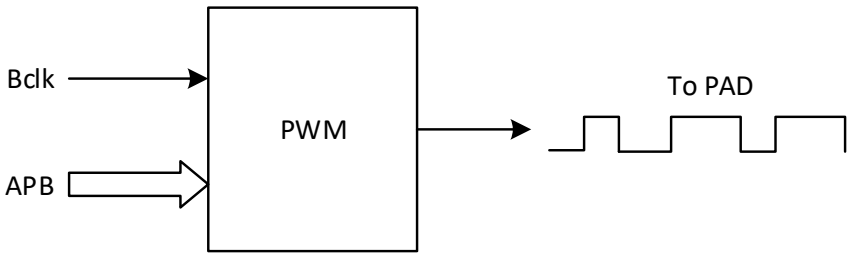


Figure 3-249 Frequency Division Generation

3.12.10.4.1.2Data Sequence Generation

The PWM module generates a waveform by data sequence generation in the PWM FIFO, memory and random modes.

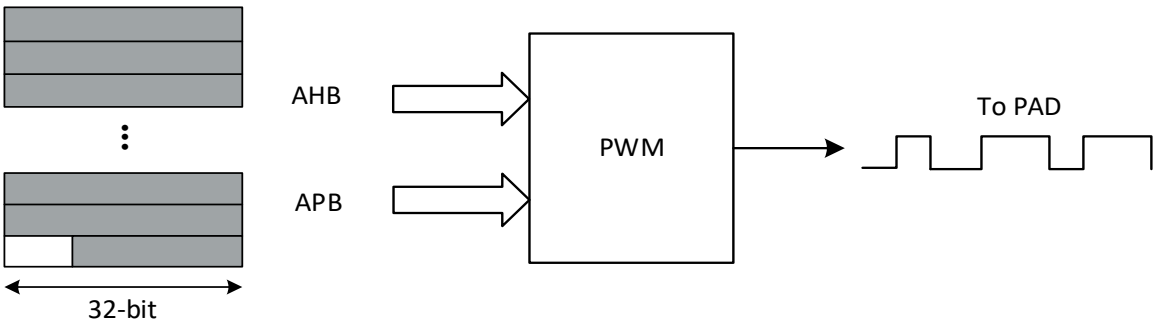


Figure 3-250 Data Sequence Generation

3.12.10.5 Theory of Operations

For the programming guide of each mode, refer to Section 3.12.10.7.

3.12.10.5.1 Normal Mode

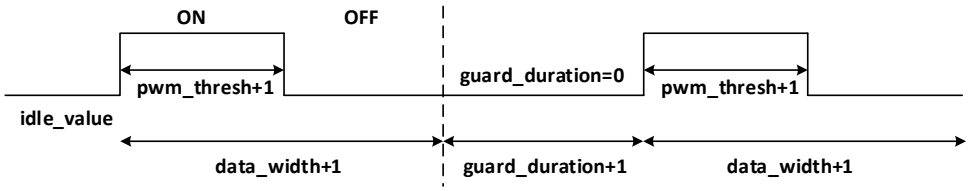


Figure 3-251 PWM Normal Mode

Figure 3-251 shows the PWM waveform in the normal mode.

The frequency is determined by PWM_DATA_WIDTH (pwm Base address+0x003C) [12:0] and CLKDIV (pwm Base address+0x0010) [2:0].

$$\text{PWM Frequency} = \frac{CLKSRC}{(2^{CLKDIV}) * (DATA\_WIDTH + 1)}$$

The duty cycle is determined by PWM_THRESH (pwm Base address+0x0040) [12:0]. Without consideration of GUARD_DURATION (pwm Base address+0x001C) [15:0], the duty cycle is:

$$\text{Duty cycle} = \frac{PWM\_THRESH + 1}{DATA\_WIDTH + 1}$$

Guard_ duration (pwm Base address+0x001C) [15:0] is the time interval between two complete waveforms. When PWM_WAVE_NUM (pwm Base address+0x0038) [15:0] = 0, it means that hardware is continuously outputting the waveform, and the waveform can only be terminated by disabling PWM. Note that CLKSRC is the frequency of the source clock.

3.12.10.5.2 FIFO Mode

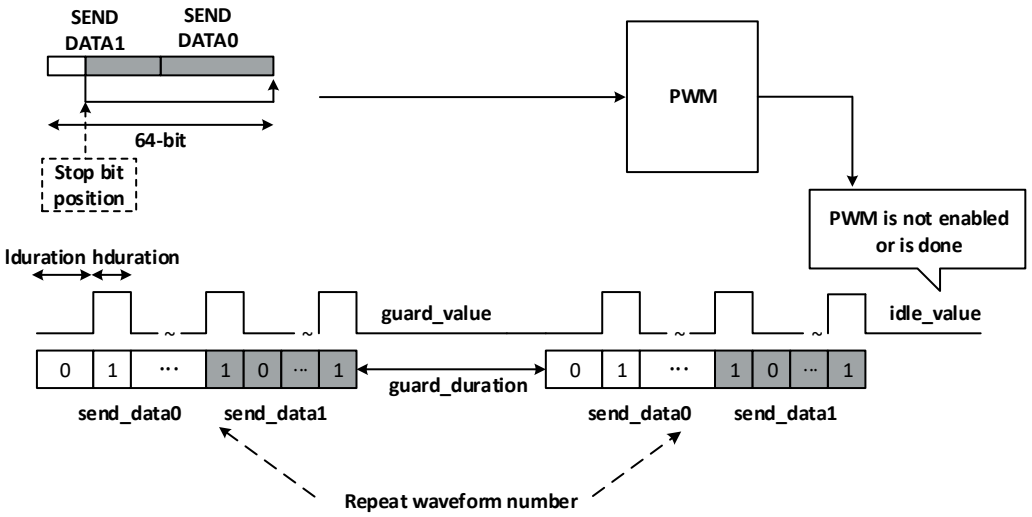


Figure 3-252 FIFO Mode

If the pulse sequence data is less than or equal to 64-bit, the data can be directly set in PWM_SEND_DATA0 (pwm Base address+0x0030) [31:0], PWM_SEND_DATA1 (pwm Base address+0x0034) [31:0] and SRCSEL (pwm Base address+0x0010) [5] =0 to reduce memory bandwidth, where SRCSEL is set to 0 to indicate that PWM is in FIFO mode. STOP_BITPOS (pwm Base address+0x0010) [14:9] is used to indicate the stop bit position in the total 64-bit data.

For example, if STOP_BITPOS (pwm Base address+0x0010) [14:9] is 31, only PWM_SEND_DATA0 (pwm Base address+0x0030) [31:0] is generated. And if STOP_BITPOS (pwm Base address+0x0010) [14:9] is 63, PWM_SEND_DATA0 (pwm Base address+0x0030) [31:0] and PWM_SEND_DATA1 (pwm Base address+0x0034) are generated.

3.12.10.5.3 Memory Mode

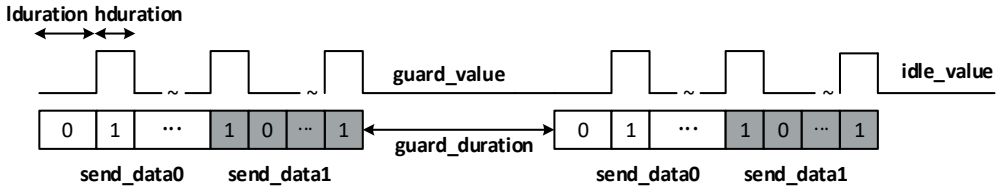


Figure 3-253 Memory Mode

In the periodical mode, all pulse sequences are repeatedly generated by the number of PWM_WAVE_NUM (pwm Base address+0x0038) [15:0]. If PWM_WAVE_NUM (pwm Base address+0x0038) [15:0] =0, the hardware continuously outputs waveforms, and the waveform generation can be stopped by PWM_ENABLE (pwm Base address+0x0000) [31:0].

SRCSEL (pwm Base address+0x0010) [5] =1 means the memory mode. The pulse sequence data is put in memory with address set by PWM_BUF0_BASE_ADDR (pwm Base address+0x0020) [31:0] and the length is PWM_BUF0_SIZE (pwm Base address+0x0024) [15:0]. STOP_BITPOS (pwm Base address+0x0010) [14:9] indicates the stop bit position in the last 32-bit data.

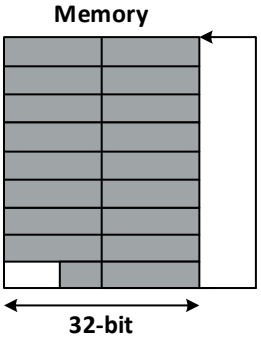


Figure 3-254 Memory Mode and Stop-bit Position

Note:
It can be any kind of memory (usually SYSRAM) so long as the software can read or write the address. The software requests to locate an unused memory and get the address, so it is necessary to write the address and length at PWM_BUF0_BASE_ADDR (pwm Base address+0x0020) and PWM_BUF0_SIZE (pwm Base address+0x0024).

3.12.10.5.4 Random Mode

On the other hand, the pulse sequence is stored in dual memory buffers in random mode. Figure 3-255 shows the format of pulse sequences stored in the memory.

The valid bit indicates that the data is ready in the respective memory buffer. The PWM generation clears this bit after all data in that buffer is fetched. The memory buffers are set by the address PWM_BUF0_BASE_ADDR (pwm Base address+0x0020) [31:0] and PWM_BUF0_SIZE (pwm Base address+0x0024) [15:0] for memory 0, and PWM_BUF1_BASE_ADDR (pwm Base address +0x0028) [31:0] and PWM_BUF1_SIZE (pwm Base address+0x002C) [15:0] for memory 1.

The program should prepare for the pulse sequence and set the valid bit to 1 in time before all data in other memory buffers is fetched. Otherwise, the hardware issues an UNDERFLOW interrupt to inform that the pulse generation stops because there is no valid data. If an UNDERFLOW interrupt is received, the software needs to disable PWM, set the valid bit again and enable PWM to restart pulse generation.

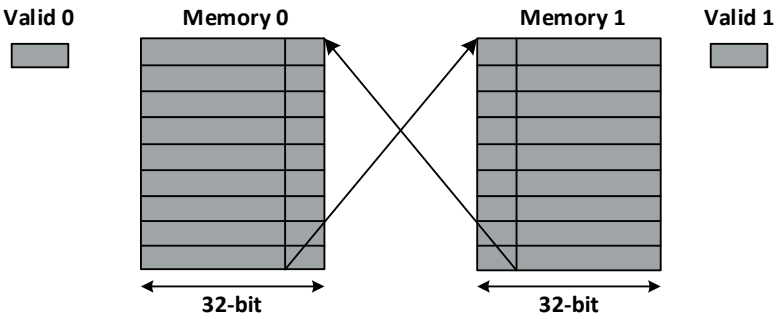


Figure 3-255 PWM Random Mode

Note:
Any kind of memory can be used by PWM (usually SYSRAM) so long as the software can read or write the address. The software issues a request, and PWM can locate an unused memory and get the address. So, you need to assign the address and length at PWM_BUF0_BASE_ADDR, PWM_BUF0_SIZE, PWM_BUF1_BASE_ADDR and PWM_BUF1_SIZE.

3.12.10.6 PWM Signal Descriptions

Table 3-202 presents PWM signal descriptions.

Table 3-202 PWM Signal Descriptions

Signal Name	Type	Description	Ball Location
PWM_0	DO	PWM output 0	D36, AL16, AM8
PWM_1	DO	PWM output 1	D37, AK16, AT10
PWM_2	DO	PWM output 2	E36, AT16, AN10
PWM_3	DO	PWM output 3	E35, AN15, AT11

3.12.10.7 Programming Guide

3.12.10.7.1 Normal Mode

Table 3-203 Normal Mode Setting Procedures

PWM Setting Sequence for Old Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
1	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b0	Disable PWM[N]
2	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[15]	OLD_PWM_MODE	1'b1	
3	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[8]	GUARD_VALUE	USER_DEFINED	
4	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[7]	IDLE_VALUE	USER_DEFINED	
5	Set PWM_WAVE_NUM	W	PWM_BASE + 0x038+ PWM_NUM*0x40	[15:0]	PWM_WAVE_NUM	USER_DEFINED	If WAVE_NUM=0, waveform generation does not stop until it is disabled.
6	Set PWM_GDURATION	W	PWM_BASE + 0x01C+ PWM_NUM*0x40	[15:0]	PWM_GDURATION	USER_DEFINED	
7	Set PWM_DATA_WIDTH	W	PWM_BASE + 0x3C + PWM_NUM*0x40	[12:0]	PWM_DATA_WIDTH	USER_DEFINED	
8	Set PWM_THRESH	W	PWM_BASE + 0x40 + PWM_NUM*0x40	[12:0]	PWM_THRESH	USER_DEFINED	
9	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b1	Enable PWM[N]

3.12.10.7.2 FIFO Mode

Table 3-204 FIFO Mode Setting Procedures

PWM Setting Sequence for FIFO Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
1	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b0	Disable PWM[N]
2	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[15]	OLD_PWM_MODE	1'b0	
3	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[6]	MODE	1'b0	
4	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[5]	SRCSEL	1'b0	
5	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[8]	GUARD_VALUE	USER_DEFINED	
6	Set IDLE_VALUE	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[7]	IDLE_VALUE	USER_DEFINED	
7	Set PWM_WAVE_NUM	W	PWM_BASE + 0x038+ PWM_NUM*0x40	[15:0]	PWM_WAVE_NUM	USER_DEFINED	If WAVE_NUM=0, waveform generation does not stop until it is disabled.
8	Set PWM_GDURATION	W	PWM_BASE + 0x01C+ PWM_NUM*0x40	[15:0]	PWM_GDURATION	USER_DEFINED	
9	Set PWM_HDURATION	W	PWM_BASE + 0x014+ PWM_NUM*0x40	[15:0]	PWM_HDURATION	USER_DEFINED	
10	Set PWM_LDURATION	W	PWM_BASE + 0x018+ PWM_NUM*0x40	[15:0]	PWM_LDURATION	USER_DEFINED	
11	Set PWM_SEND_DATA0	W	PWM_BASE + 0x030+ PWM_NUM*0x40	[31:0]	PWM_SEND_DATA0	USER_DEFINED	This value should be written only in periodical FIFO mode. In other modes, this buffer is for an internal

PWM Setting Sequence for FIFO Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
							memory access.
12	Set PWM_SEND_DATA1	W	PWM_BASE + 0x034+ PWM_NUM*0x40	[31:0]	PWM_SEND_DATA1	USER_DEFINED	This value should be written only in periodical FIFO mode. In other modes, this buffer is for internal memory access.
13	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[14:9]	STOP_BITPOS	USER_DEFINED	Stop bit position for source data in periodical mode In FIFO mode, it is used to indicate the stop bit position in a total of 64 bits. In memory mode, it is for the stop bit position of the last 32 bits.
14	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b1	Enable PWM[N]

3.12.10.7.3 Memory Mode

Table 3-205 Memory Mode Setting Procedures

PWM Setting Sequence for Memory Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
1	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b0	Disable PWM[N]
2	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[15]	OLD_PWM_MODE	1'b0	
3	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[6]	MODE	1'b0	

PWM Setting Sequence for Memory Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
4	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[5]	SRCSEL	1'b1	
5	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[8]	GUARD_VALUE	USER_DEFINED	
6	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[7]	IDLE_VALUE	USER_DEFINED	
7	Set PWM_WAVE_NUM	W	PWM_BASE + 0x038+ PWM_NUM*0x40	[15:0]	PWM_WAVE_NUM	USER_DEFINED	If WAVE_NUM=0, waveform generation does not stop until it is disabled.
8	Set PWM_GDURATION	W	PWM_BASE + 0x01C+ PWM_NUM*0x40	[15:0]	PWM_GDURATION	USER_DEFINED	
9	Set PWM_HDURATION	W	PWM_BASE + 0x014+ PWM_NUM*0x40	[15:0]	PWM_HDURATION	USER_DEFINED	
10	Set PWM_LDURATION	W	PWM_BASE + 0x018+ PWM_NUM*0x40	[15:0]	PWM_LDURATION	USER_DEFINED	
11	Set PWM_BUF0_BASE_ADDR	W	PWM_BASE + 0x020+ PWM_NUM*0x40	[31:0]	PWM_BUF0_BASE_ADDR	USER_DEFINED	Base address of memory buffer0 for PWM's waveform data
12	Set PWM_BUF0_SIZE	W	PWM_BASE + 0x024+ PWM_NUM*0x40	[15:0]	PWM_BUF0_BASE_ADDR_EXTEND	USER_DEFINED	Length of waveform data in memory buffer0 PWM should generate
13	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[14:9]	STOP_BITPOS	USER_DEFINED	Stop bit position for source data in periodical mode. In FIFO mode, it is used to indicate the stop bit position in a total of 64 bits. In memory mode, it is for

PWM Setting Sequence for Memory Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
							the stop bit position of the last 32 bits.
14	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b1	Enable PWM[N]

3.12.10.7.4 Random Mode

Table 3-206 Random Mode Setting Procedures

PWM Setting Sequence for Random Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
1	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b0	Disable PWM[N]
2	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[15]	OLD_PWM_MODE	1'b0	
3	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[6]	MODE	1'b1	
4	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[5]	SRCSEL	1'b1	
5	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[8]	GUARD_VALUE	USER_DEFINED	
6	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[7]	IDLE_VALUE	USER_DEFINED	
7	Set PWM_GDURATION	W	PWM_BASE + 0x01C+ PWM_NUM*0x40	[15:0]	PWM_GDURATION	USER_DEFINED	
8	Set PWM_HDURATION	W	PWM_BASE + 0x014+ PWM_NUM*0x40	[15:0]	PWM_HDURATION	USER_DEFINED	
9	Set PWM_LDURATION	W	PWM_BASE + 0x018+ PWM_NUM*0x40	[15:0]	PWM_LDURATION	USER_DEFINED	
10	Set PWM_BUF0_BASE_ADDR	W	PWM_BASE + 0x020+ PWM_NUM*0x40	[31:0]	PWM_BUF0_BASE_ADDR	USER_DEFINED	Base address of memory buffer0 for PWM's waveform data

PWM Setting Sequence for Random Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
11	Set PWM_BUF0_SIZE	W	PWM_BASE + 0x024+ PWM_NUM*0x40	[15:0]	PWM_BUF0_BASE_ADDR_EXTEND	USER_DEFINED	Length of waveform data in memory buffer0 PWM should generate
12	Set PWM_BUF1_BASE_ADDR	W	PWM_BASE + 0x028+ PWM_NUM*0x40	[31:0]	PWM_BUF1_BASE_ADDR	USER_DEFINED	Base address of memory buffer1 for PWM's waveform data
13	Set PWM_BUF1_SIZE	W	PWM_BASE + 0x02C+ PWM_NUM*0x40	[15:0]	PWM_BUF1_BASE_ADDR_EXTEND	USER_DEFINED	Length of waveform data in memory buffer1 PWM should generate
14	Set PWM_VALID	W	PWM_BASE + 0x048+ PWM_NUM*0x40	[1:0]	BUF1_VALID/ BUF0_VALID	2'b11	Memory1/0 is not empty. When finishing writing data to memory1, write 1 to inform PWM the data in memory1 is ready.
15	Set PWM_CON	W	PWM_BASE + 0x010+ PWM_NUM*0x40	[14:9]	STOP_BITPOS	USER_DEFINED	Stop bit position for source data in periodical mode. In FIFO mode, it is used to indicate the stop bit position in a total of 64 bits. In memory

PWM Setting Sequence for Random Mode							
Step	**Description**	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
							mode, it is for the stop bit position of the last 32 bits.
16	Set PWM_ENABLE	W	PWM_BASE + 0x0000	[N]	PWM_ENABLE	1'b1	Enable PWM[N]

3.12.10.8 Register Definition

Refer to "*MT8395 Register Map*" for detailed register descriptions.

3.13 Miscellaneous

3.13.1 Timers and Counters

3.13.1.1 System Timer (SYSTMR)

3.13.1.1.1 Overview

The SYSTMR is a 64-bit and always-on up-counter used as a universal timer in the device. The counter value of SYSTMR is passed to the application cores (A78 and A55), SCP, GPU, and other processors to provide uniform system timestamps for operating systems such as Android™, Linux®, and RTOS.

3.13.1.1.2 Feature

The SYSTMR supports the following key features:

- Enabled by default to tick with 13 MHz clock period
- Hardware counter incremented compensation when switching to 32 kHz clock source
- 8 × 32-bit counter timeout value (read as 32-bit down counter)
- Security access permission control for each control register (with one-time lock bit)

3.13.1.1.3 Block Diagram

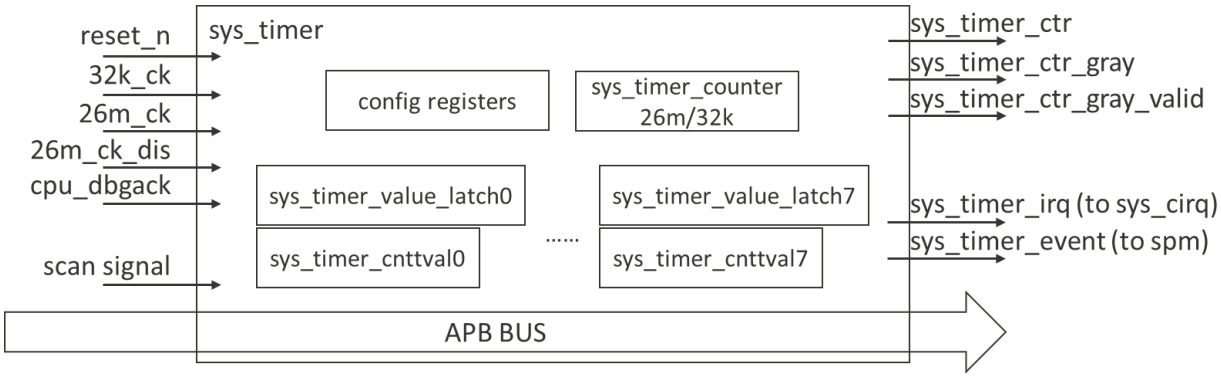


Figure 3-256 Block Diagram of SYSTMR

3.13.1.1.4 Function Description

- Low power mode
  When the system enters low power mode, SYSTMR will receive 26m_ck_dis from SPM and switch to 32 kHz clock. SYSTMR can ensure the accuracy of counter value in low power mode by using the 32 kHz clock counter.
- Halt on debug
  If the CPU wants to stop system debugging, the CPU can send the cpu_dbgack signal. SYSTMR will stop counting after receiving cpu_dbgack to keep the current counter value.
- Timer IRQ
  You can set the timer value by writing to the CNTTVAL register. Then, the timer starts counting down. The timer issues an IRQ when it counts down to 0.

3.13.1.1.5 Theory of Operations

SYSTMR is a general counter with the function of counting up and counting down. However, in low power mode, we use a compensation mechanism to ensure that the exact counter value can be maintained between 26m clock and 32k clock. The relationship between 26 MHz and 32 kHz is not an integer, so there is a Quantization Error.

1. 26 MHz/32.768 kHz =793.4570313. One cycle of 32.768 kHz will take 793.4570313 cycles of 26 MHz.
2. If we add the integer 793, each cycle of 32 kHz will cause 0.45703125 cycles of 26 MHz error (about 17.5781 ns).
3. One cycle of 32 kHz produces an error of 17.5781 ns, 56.8889M cycles of 32k produces an error of about 1 second -> the time to accumulate a 1 second error is 1736.11 seconds.

Is it mean every 1736.11 sec pass, there is 1 sec error. In order to eliminate the Quantization Error, we conduct the following analysis:

1. Using moving average find 793.45703125 => 793.45703125 = 793*X + 794*(1-X) => X= 0.54296875.
2. We search the best P and Q by Numerical Analysis, we know 793*(139/256) + 794*(117/256) = 793.45703125.

Bit Counter	P/Q	X(=P/Q)	Eq. Increment	Error	Day	Year
5	13/24	0.541666667	793.4583333	-5.00801E-11	-7.0529514	-0.02
6	19/35	0.542857143	793.4571429	-4.29258E-12	-82.284433	-0.23

Bit Counter	P/Q	X(=P/Q)	Eq. Increment	Error	Day	Year
7	19/35	0.542857143	793.4571429	-4.29258E-12	-82.284433	-0.23
8	120/211	0.542986425	793.4570136	-6.79821E-13	519.56742	1.42
9	139/256	0.54296875	793.4570313	0	#DIV/0!	#DIV/0!

In every 256 32k cycles, add 139 times 793 and 117 times 794 to realize the compensation mechanism.

3.13.1.1.6 Programming Guide

3.13.1.1.6.1 Counter

Although SYSTMR uses a 26 MHz clock source, the counter value is updated at a frequency of 13M. The counter value of SYSTMR is about 76.92 ns per unit. When running at 26 MHz, the counter value increases by 1; when running at 32 kHz, the counter value increases by 396 or 397.

In the CNTCR register, the counter is mainly controlled by the following bits. For a more complete register introduction, refer to "*MT8395 Register Map*".

Register Name	Address	Bit	Name	Type	Description
CNTCR	0x00000	0	CNT_EN	RW	Enable SYSTMR counter. Default setting is 1.
		2	CNT_32K_AS_EN	RW	Enable 32 kHz auto-switch function of SYSTMR counter. Default setting is 1.
		10	COMP_15_EN	RW	v1.5 is based on 13 MHz and 32 kHz to tick SYSTMR.
		11	COMP_20_EN	RW	v2.0 is only based on 32 kHz to tick SYSTMR.
		12	COMP_25_EN	RW	v2.5 solves discontinuity issue of v2.0. v2.5 and v2.0 need to be enabled together. v2.0 can be enabled alone.
		13	KP_LSB9b_EN	RW	The 10 LSB remain 0 when the clock switches to 32 kHz if this bit is set.

- CNT_32K_AS_EN
  This bit decides when 26 MHz is turned off, whether SYSTMR is switched to update the counter at 32 kHz frequency. If this bit is not enabled, the counter will pause when 26 MHz is turned off. In general, this bit should not be disabled.

- COMP_15_EN
  In compensation v1.5, SYSTMR will only use one hardware counter to count the counter value. At 26 MHz clock, the counter will add 1 every cycle; when 26 MHz is turned off, SYSTMR will use 32 kHz clock, as the discussion of 1.5 Low Power, add 396 or 397 every cycle.

- COMP_20_EN/COMP_25_EN
  In compensation v2.0, SYSTMR uses 26 MHz counter and 32 kHz counter to count the counter values, and the two counters count at 26 MHz and 32 kHz clocks, respectively. SYSTMR is mainly based on the 32 kHz counter, because the 32 kHz counter is always counting and will not be turned off.

Every time the 32 kHz clock rises, the 26 MHz counter will be aligned with the 32 kHz counter, and then the 26 MHz counter value will continue to increment at 26 MHz frequency.

In normal mode, SYSTMR will output 26 MHz counter value. Turning off the 26 MHz clock means low power mode is enabled. 26 MHz counter will pause and SYSTMR will switch to output the 32 kHz counter value.

However, this architecture may result in discontinuous SYSTMR values. For systems that are sensitive to continuous count values, the software must enable COMP_25_EN to solve the discontinuity issue.

The following describes the reading and writing of counter value.

Register Name	Address	Type	Description
CNTCV_L	0x00008	RW	64-bit SYSTMR counter value[31:0].
CNTCV_H	0x0000C	RW	64-bit SYSTMR counter value[63:32].

- Initial counter value
  Write 0 to CNTCV_H and CNTCV_L. It takes about 3 32 kHz cycles to synchronize (~100 us).
  *If CNT_EN = 0, CNTCV_H/CNTCV_L will still change. Wait for CNT_EN = 1, and then wait for 3 32 kHz cycles to update.
- Read counter value
  Because the data is from the SYSTMR clock domain to the APB clock domain, it needs to be synchronized. Due to hardware limitations, no matter where the counter value is read from, CNTCV_L needs to be read first to avoid unexpected results.

3.13.1.1.6.2 Timer

The software can control timer and interrupt through the following register.

Register Name	Address	Bit	Name	Type	Description
CNTTVAL[N]_CON	0x00040+8*N	0	CNTTVAL[N]_EN	RW	Enable system timer N.
		1	IRQ[N]_EN	RW	Enable system timer N interrupt.
		4	IRQ[N]_STA	RW	Read: system timer N interrupt status. Write: write 1'b1 to clear interrupt.

The software can set the timer value through the following register.

Register Name	Address	Type	Description
CNTTVAL[N]	0x00044+8*N	RW	System timer N down counter. Read: 32-bit down counter value. Write: set timeout value.

Note: N = 0 to 7

- Set timer value
  a. Set CNTTVAL[N]*_CON to 0x1: Enable timer[N] to write.
  b. Write counter down value to CNTTVAL[N].
  c. Set CNTTVAL[N]_CON to 0x3: Enable IRQ_EN of timer[N] for timeout IRQ.
  d. Read CNTTVAL[N]: Read current count down value (optional).

e. After IRQ is asserted, you need to write CNTTVAL[N]_CON bit 4 to clear IRQ. (CNTTVAL[N]_EN must be 1 to clear the IRQ)

- Timer domain check
  Domain check will limit the timer so that only specific domains can access it.

Register Name	Address	Bit	Name	Type	Description
DOMAIN_DIS	0x00080	7:0	RD_DOMAIN_DIS	RW	Disable domain check for read control registers of timers. Bits 7 to 0 are for timers 7 to 0. 0: Enable domain check for reading (Default). 1: Disable domain check for reading.
		15:8	WR_DOMAIN_DIS	RW	Disable domain check for write control registers of timers. Bits 15 to 8 are for timers 7 to 0. 0: Enable domain check for writing (Default). 1: Disable domain check for writing.
DOMAIN_VAL	0x0084	[3+4*N:4*N]	DOMAIN_VAL_[N]	RW	Domain value of timer N. Domain value is 4 bits.

3.13.1.2 General-Purpose Timer (APXGPT)

3.13.1.2.1 Overview

The Application Processor General-Purpose Timer (APXGPT) module consists of five sets of GPTs that feature 32-bit up-counters and one set of GPTs that contain 64-bit up-counters. Section 3.13.1.2.3 shows the block diagram. Each GPT supports four operation modes as follows. For further details, see Table 3-207 and Section 3.13.1.2.6.

- ONE-SHOT mode
- REPEAT mode
- KEEP-GO mode
- FREERUN mode

Each GPT can operate on either of the two clock sources: 32.768 kHz Real-Time Clock (RTC) or the 13 MHz system clock, with a 4-bit pre-scaler that provides a programmable clock frequency from these two clock sources.

By selecting the desired modes and working clocks, a specific time delay can be obtained. Upon reaching a programmable timer value, an Interrupt Request (IRQ) will be sent to the CPU and System Power Management (SPM).

3.13.1.2.2 Features

- Selection of two clock sources for the up-counter of each GPT
- Programmed to be active in the low power mode
- Interrupt generation when a programmable timer value is reached
- Three comparison modes available for the set timer boundary value

3.13.1.2.3 Block Diagram

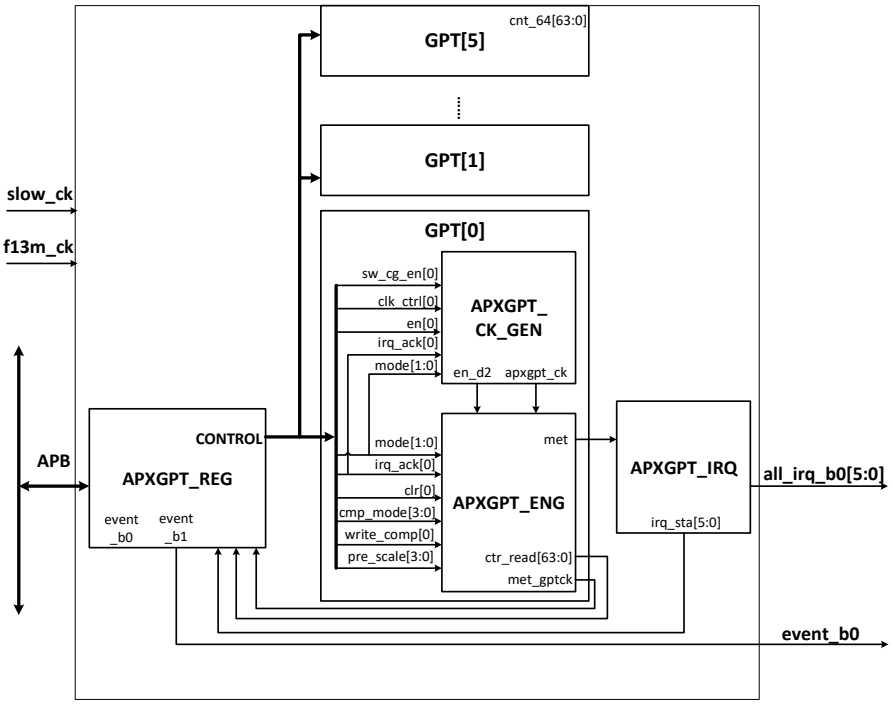


Figure 3-257 Block Diagram of APXGPT

3.13.1.2.4 Function Description

The APXGPT is primarily built upon an up-counter that counts using the rising edge of a clock. The operating clock frequency can be selected from two sources, usually 13 MHz and 32.768 kHz, which can be internally divided to provide additional frequency options. As the clock period is known, the counter boundary should be set to obtain a specific time delay. The time is then calculated by multiplying the count by the clock period. Upon reaching the counter boundary, an interrupt request will be generated.

The table below presents detailed descriptions for the four operation modes of each GPT.

Table 3-207 Operation Mode of GPT

Mode	Auto Stop	Interrupt Supported	Count Behavior	When GPTn_COUNT Equals to GPTn_COMPARE	Example: Compare is set to 2 (Underlining means that an interrupt is asserted.)
ONE-SHOT	Yes	Yes	Count stops when GPTn_COUNT equals GPTn_COMPARE	EN is reset to 0	0,1,**2**,2,2,2,2,2,2,2,2,2,…
REPEAT	No	Yes	Count is reset to 0 when GPTn_COUNT equals GPTn_COMPARE	Count is reset to 0	0,1,**2**,0,1,**2**,0,1,**2**,0,1,**2**…
KEEP-GO	No	Yes	Count is reset to 0 when the count is overflowed.	No action	0,1,**2**,3,4,5,6,7,8,9,10,…
FREERUN	No	No	Count is reset to 0 when the count is overflowed.	No action	0,1,2,3,4,5,6,7,8,9,10,…

Note:

- GPTn_COUNT (APXGPT Base address+(0x0008+0x20*(n-1))), GPTn_COMPARE (apxgpt Base address + (0x000C+0x20*(n-1))) (n=0,1,2,3,4,5)
- Each timer operates independently and can be programmed to select between the 32.768 kHz RTC or the13 MHz system clock. Once the clock source is set, the selected clock's division ratio can be programmed and fine-tuned within the range of 1 to 13, or coarse-tuned with values of 16, 32 and 64.

3.13.1.2.4.1 Clocking

The APXGPT module employs four main input clocks, namely:

- slow_clock
- f13m_ck
- pclk_ck
- hclk_ck

The first two clocks serve as the operation clock of the up-counter. Select one of them through the register setting. The APB clock (PCLK) and hopping clock (HCLK) are bus-related clocks.

Table 3-208 Clock Source of GPT

Name	Frequency	Description
slow_ck	32.768 kHz	APXGPT counter clock
f13m_ck	13 MHz	APXGPT counter clock
pclk_ck	26 MHz	APB clock
hclk_ck	26 MHz	To synchronize with the bus clock

3.13.1.2.4.2 Interrupts

When the GPT reaches the programmable compared value of the up-counter, "GPTn" (n = 0, 1, 2, 3, 4, 5) generates an interrupt request (IRQ) to send to the GPU. These GPT interrupts are merged and connected to the SPM.

Upon GPT triggering an IRQ, it issues a wake-up signal to the "Sleep Control" to wake up the MCU if it is in the sleep mode.

3.13.1.2.5 Theory of Operations

The read operation value of the GPT6 64-bit timer is split into two 32-bit APB reads. The lower word is read first, followed by the higher word. The read operation of lower word freezes the "read value" of the higher word, but does not freeze the timer counting. This ensures that the separated read operation acquires the correct timer value.

When programming and utilizing the GPT, note the following points:

- The counter value can be read at any time when the clock source is the system clock or the RTC.
- The comparative value can be programmed at any time. If it is rewritten during a count operation, the counter is reset to 0 and restarts the count.

3.13.1.2.6 Programming Guide

The following sections describe the operating sequence of all the operation modes, using GPT0 as an example to demonstrate the programming sequence.

3.13.1.2.6.1 ONE-SHOT Mode

Table 3-209 APXGPT ONE-SHOT Mode Setting Flow

Step	Address	Register Name	Local Address	R/W	Value	Description
Set the APXGPT control register.						
1	APXGPT base address + 0x0000	GPT0_CON	CLK	RW	Refer to the description.	Select GPT0 clock source, 00: System clock (13 MHz) 01: RTC (32.768 kHz)
2	APXGPT base address + 0x0000	GPT0_CON	CLKDIV	RW	Refer to the description.	Set the GPT0 clock divider, 0000: Clock source divided by 1 0001: Clock source divided by 2 0010: Clock source divided by 3 0011: Clock source divided by 4 0100: Clock source divided by 5 0101: Clock source divided by 6 0110: Clock source divided by 7 0111: Clock source divided by 8 1000: Clock source divided by 9 1001: Clock source divided by 10 1010: Clock source divided by 11 1011: Clock source divided by 12 1100: Clock source divided by 13 1101: Clock source divided by 16 1110: Clock source divided by 32 1111: Clock source divided by 64
3	APXGPT base address + 0x0000	GPT0_CON	MODE	RW	2'b00	Set the GPT0 control register for ONE-SHOT mode. 00: ONE-SHOT mode 01: REPEAT mode 10: KEEP-GO mode 11: FREERUN mode
4	APXGPT base address + 0x000C	GPT0_COMPARE	COMPARE	RW	Refer to the description.	Set the compare value of GPT0.
5	APXGPT base address + 0x0000	GPT0_CON	IRQEN	RW	1'b1	Set IRQEN to “1” to enable GPT0 interrupt.
Enable GPT0 and wait for GPT0 interrupt.						
6	APXGPT base address + 0x0000	GPT0_CON	EN	RW	1'b1	Set EN to “1” to enable GPT0.
7	APXGPT base address + 0x0000	GPT0_CON	IRQSTA	RO	Refer to the description.	Wait for IRQSTA = “1”, 0: No associated interrupt is generated. 1: Associated interrupt is pending and waiting for service.
Clear GPT0 interrupt (Choose one of the two methods)						

Step	Address	Register Name	Local Address	R/W	Value	Description
8	APXGPT base address + 0x0000	GPT0_CON	PREFIX	WO	4'h2	Set PREFIX= 4'h2 to clear GPT0 interrupt.
	APXGPT base address + 0x0FC0	IRQ_STA	IRQ_STA	WO	1'b1	Interrupt of GPTn. Can be written to 1 to clear.

3.13.1.2.6.2 REPEAT Mode

Table 3-210 APXGPT REPEAT Mode Setting Flow

Step	Address	Register name	Local address	R/W	Value	Description
Set the APXGPT control register.						
1	APXGPT base address + 0x0000	GPT0_CON	CLK	RW	Refer to the description.	Select GPT0 clock source, 00: System clock (13 MHz) 01: RTC (32.768 kHz)
2	APXGPT base address + 0x0000	GPT0_CON	CLKDIV	RW	Refer to the description.	Set the GPT0 clock divider, 0000: Clock source divided by 1 0001: Clock source divided by 2 0010: Clock source divided by 3 0011: Clock source divided by 4 0100: Clock source divided by 5 0101: Clock source divided by 6 0110: Clock source divided by 7 0111: Clock source divided by 8 1000: Clock source divided by 9 1001: Clock source divided by 10 1010: Clock source divided by 11 1011: Clock source divided by 12 1100: Clock source divided by 13 1101: Clock source divided by 16 1110: Clock source divided by 32 1111: Clock source divided by 64
3	APXGPT base address + 0x0000	GPT0_CON	MODE	RW	2'b01	Set the GPT0 control register for REPEAT mode. 00: ONE-SHOT mode 01: REPEAT mode 10: KEEP-GO mode 11: FREERUN mode
4	APXGPT base address + 0x000C	GPT0_COMPARE	COMPARE	RW	Refer to the description.	Set the compare value of GPT0.
5	APXGPT base address + 0x0000	GPT0_CON	IRQEN	RW	1'b1	Set IRQEN= “1” to enable GPT0 interrupt.
Enable GPT0 and wait for GPT0 interrupt						
6	APXGPT base address + 0x0000	GPT_CON	EN	RW	1'b1	Set EN to “1” to enable GPT0.
7	APXGPT base address + 0x0000	GPT0_CON	IRQSTA	RO	Refer to the description.	Wait for IRQSTA = “1”, 0: No associated interrupt is generated. 1: Associated interrupt is pending and waiting for service.
Clear GPT0 interrupt (Choose one of the two methods)						

Step	Address	Register name	Local address	R/W	Value	Description
8	APXGPT base address + 0x0000	GPT0_CON	PREFIX	WO	4'h2	Set PREFIX= 4'h2 to clear GPT0 interrupt.
	APXGPT base address + 0x0FC0	IRQ_STA	IRQ_STA	WO	1'b1	Interrupt of GPTn. Can be written to 1 to clear.

3.13.1.2.6.3 KEEP-GO Mode

Table 3-211 APXGPT KEEP-GO Mode Setting Flow

Step	Address	Register name	Local address	R/W	Value	Description
Set the APXGPT control register.						
1	APXGPT base address + 0x0000	GPT0_CON	CLK	RW	Refer to the description.	Select the GPT0 clock source, 00: System clock (13 MHz) 01: RTC (32.768 kHz)
2	APXGPT base address + 0x0000	GPT0_CON	CLKDIV	RW	Refer to the description.	Set the GPT0 clock divider, 0000: Clock source divided by 1 0001: Clock source divided by 2 0010: Clock source divided by 3 0011: Clock source divided by 4 0100: Clock source divided by 5 0101: Clock source divided by 6 0110: Clock source divided by 7 0111: Clock source divided by 8 1000: Clock source divided by 9 1001: Clock source divided by 10 1010: Clock source divided by 11 1011: Clock source divided by 12 1100: Clock source divided by 13 1101: Clock source divided by 16 1110: Clock source divided by 32 1111: Clock source divided by 64
3	APXGPT base address + 0x0000	GPT0_CON	MODE	RW	2'b10	Set the GPT0 control register for KEEP-GO mode. 00: ONE-SHOT mode 01: REPEAT mode 10: KEEP-GO mode 11: FREERUN mode
4	APXGPT base address + 0x000C	GPT0_COMPARE	COMPARE	RW	Refer to the description.	Set the compare value of GPT0
5	APXGPT base address + 0x0000	GPT_IRQEN	IRQEN	RW	1'b1	Set IRQEN = “1” to enable GPT0 interrupt.
Enable GPT0.						
6	APXGPT base address + 0x0000	GPT0_CON	EN	RW	1'b1	Set EN to “1” to enable GPT0.
7	APXGPT base address + 0x0000	GPT0_CON	IRQSTA	RO	Refer to the description.	Wait for IRQSTA = “1”, 0: No associated interrupt is generated, 1: Associated interrupt is pending and waiting for service.

Step	Address	Register name	Local address	R/W	Value	Description
Read the GPT0 counter value						
8	APXGPT base address + 0x0008	GPT0_COUNT	COUNTER	RO	Refer to the description.	The timer counter of GPT0.
Clear GPT0 interrupt. (Choose one of the two methods)						
9	APXGPT base address + 0x0000	GPT0_CON	PREFIX	WO	4'h2	Set PREFIX = 4'h2 to clear GPT0 interrupt.
	APXGPT base address + 0x0FC0	IRQ_STA	IRQ_STA	WO	1'b1	Interrupt of GPTn. Can be written to 1 to clear.

3.13.1.2.6.4 FREERUN Mode

Table 3-212 APXGPT FREERUN Mode Setting Flow

Step	Address	Register name	Local address	R/W	Value	Description
Set the APXGPT control register.						
1	APXGPT base address + 0x0000	GPT0_CON	CLK	RW	Refer to the description.	Select GPT0 clock source, 00: System clock (13 MHz) 01: RTC (32.768 kHz)
2	APXGPT base address + 0x0000	GPT0_CON	CLKDIV	RW	Refer to the description.	Set the GPT0 clock divider, 0000: Clock source divided by 1 0001: Clock source divided by 2 0010: Clock source divided by 3 0011: Clock source divided by 4 0100: Clock source divided by 5 0101: Clock source divided by 6 0110: Clock source divided by 7 0111: Clock source divided by 8 1000: Clock source divided by 9 1001: Clock source divided by 10 1010: Clock source divided by 11 1011: Clock source divided by 12 1100: Clock source divided by 13 1101: Clock source divided by 16 1110: Clock source divided by 32 1111: Clock source divided by 64
3	APXGPT base address + 0x0000	GPT0_CON	MODE	RW	2'b11	Set the GPT0 control register for KEEP-GO mode, 00: ONE-SHOT mode 01: REPEAT mode 10: KEEP-GO mode 11: FREERUN mode
Enable GPT0.						
4	APXGPT base address + 0x0000	GPT0_CON	EN	RW	1'b1	Set EN to “1” to enable GPT0.
Read the GPT0 counter value.						

Step	Address	Register name	Local address	R/W	Value	Description
5	APXGPT base address + 0x0008	GPT0_COUNT	COUNTER	RO	Refer to the description.	The timer counter of GPT0.

3.13.1.2.6.5 GPT5 64-Bit Counter Read

Table 3-213 GPT5 64-bit Counter Read Flow

Step	Address	Register name	Local address	R/W	Value	Description
Read the lower 32 bits of the GPT5 counter.						
1	APXGPT base address + 0x00A8	GPT5_COUNT_L		RO	Refer to the description.	The lower word of the timer count of GPT5.
Read the higher 32 bits of the GPT5 counter.						
2	APXGPT base address + 0x00B0	GPT5_COUNT_H		RO	Refer to the description.	The higher word of the timer count of GPT5.

3.13.1.2.7 Register Definition

Refer to "*MT8395 Register Map*" for detailed register descriptions.

3.13.1.3 Watchdog Timer (WDT)

The WDT module is a part of TOPRGU. For more information refer to Section 5.5 Reset.

3.13.2 PMIC Wrapper (PWRAP)

3.13.2.1 Overview

The PWRAP serves as a bridge for the communication between SoC and PMIC. Figure 3-258 shows its top block diagram. Multiple masters control or get information from the PMIC in the SoC, as Figure 3-258 shows.

PMIF is a block for user channel command collection and management. It allocates all user requests and decides which user occupies the bus to transfer user commands to the PMIC.

PMICSPI_MST is MediaTek's SPI master, which receives PMIF's commands and transfers them using MediaTek SPI protocol. PMICSPMI_MST is a System Power Management Interface (SPMI) master, which receives PMIF's commands and transfers them using the SPMI protocol.

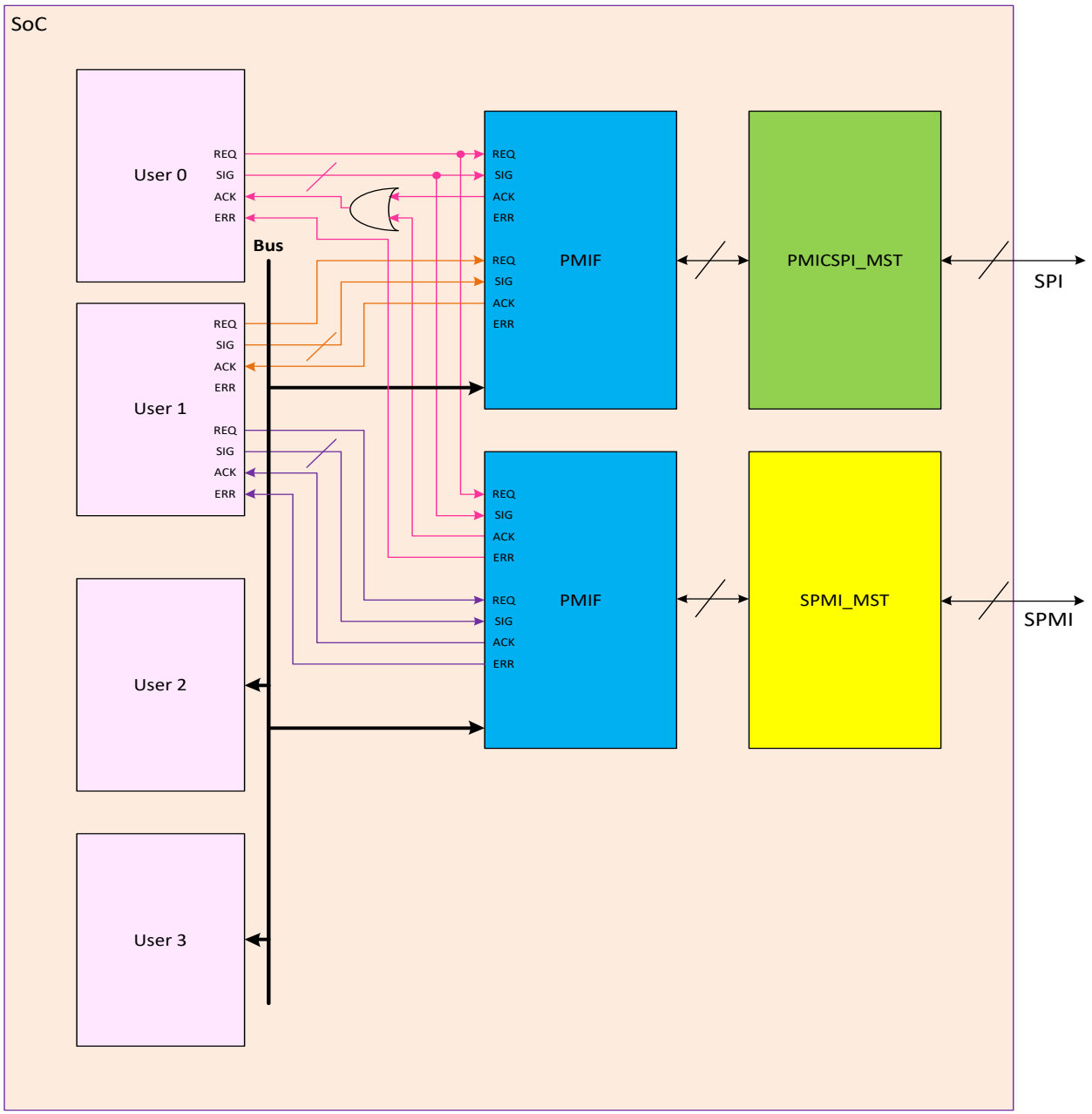


Figure 3-258 PWRAP Top Diagram

3.13.2.2 Features

- Fast auto SPI format generator for PMIC registers read/write
- APB3.0 bus lock scheme when SPI is busy
- Manual SPI format generator
- Dual I/O SPI mode
- Separated frequency between controller and SPI or SPMI
- One master with multiple request capable slaves for SPMI slaves
- Use master write commands as PMIC interrupts for SPMI slaves
- Slave operation state change between reset/sleep/shutdown/wakeup for SPMI slaves

3.13.2.3 Block Diagram

Figure 3-259, Figure 3-260 and Figure 3-261 show the block diagrams of PMIF, PMICSPI_MST and PMICSPMI_MST, respectively.

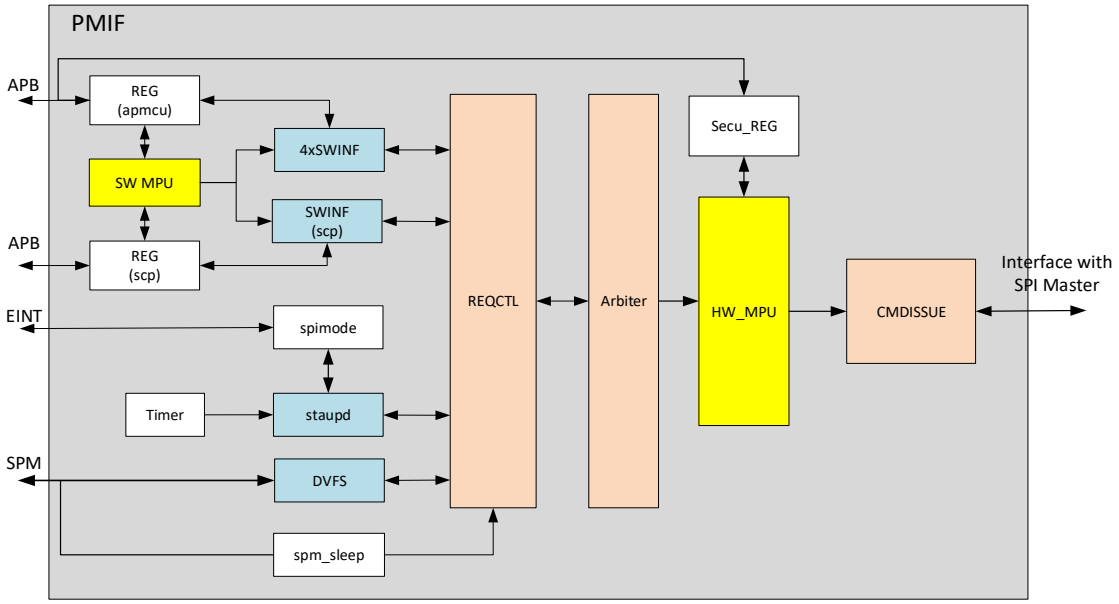


Figure 3-259 Block Diagram of PMIF

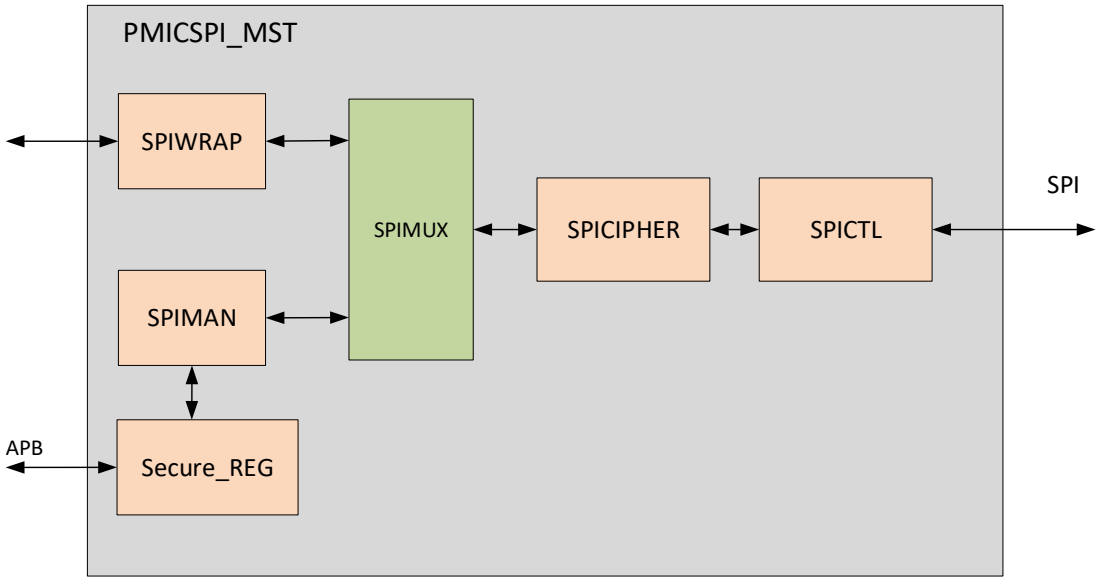


Figure 3-260 PMIFSPI_MST Block Diagram

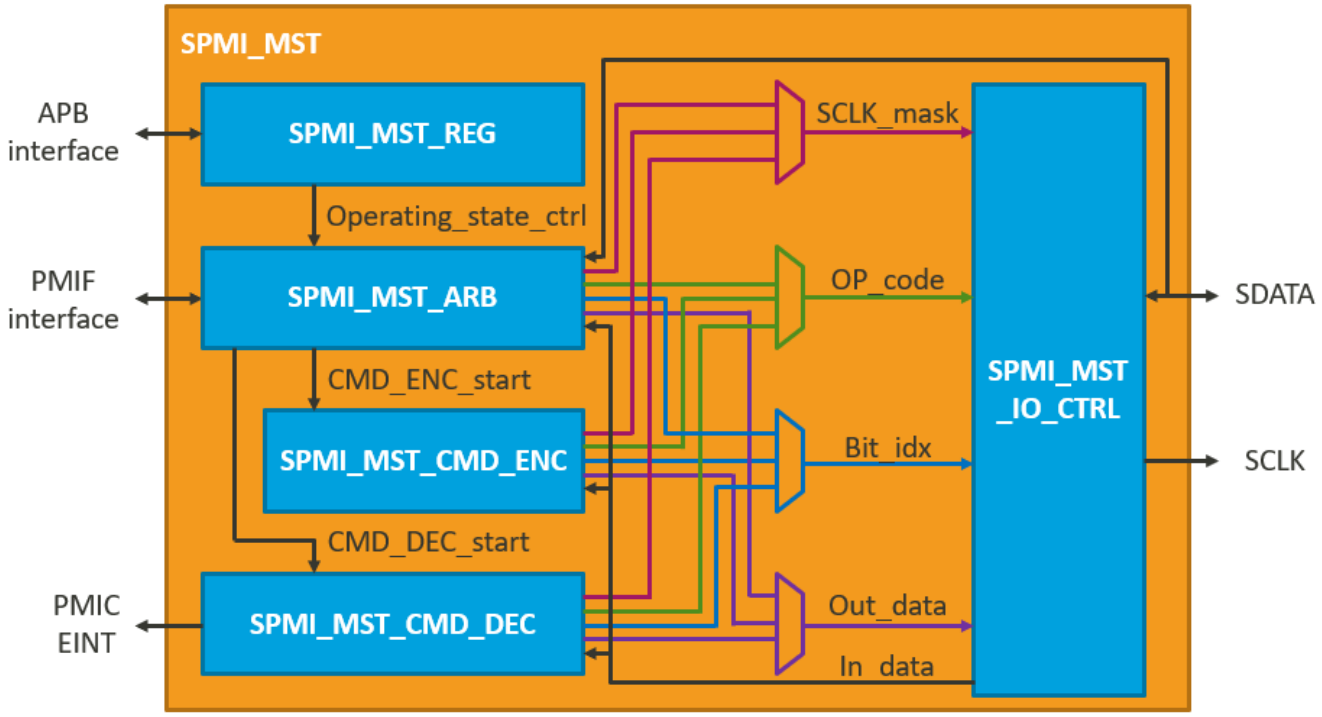


Figure 3-261 PMIFSPMI_MST Block Diagram

3.13.2.4 Function Description

User channels of PMIF consist of four APMCU SWINF (software interfaces), one SCP software interface, one SPM DVFS control interface and one PMIF internal staupd (status update) channel.

- Timer is generated periodically operation for staupd
- spimode is a mode switch function for an EINT (external interrupt) source.
  - In normal mode, EINT source is from staupd.
  - In sleep mode, EINT is obtained from the pin PWRAP_SPI0_MI.
- PWRAP_SPI0_MI is part of the SPI control interface. It can be used as a data pin or interrupt pin, depending on different system scenarios. It is used as a data pin when the system is in normal mode, and as an EINT pin when the system is in sleep mode.
- spm_sleep is a sleep mode protection module. When the SoC enters sleep mode, SPM will request pmic_wrap to gate all channel requests to REQCTL. After PRWAP finishes the REQCTL queue, it sends an acknowledgment to SPM to proceed the sleep control.
- REQCTL is user request control and it sends requests to the arbiter.
- Arbiter: Arbitration of 4 AP MCU SWINFs, 1 SCP SWINF, 1 SPM DVFS and 1 staupd chanel.
- HWMPU serves all channel operation protection.
- CMDISSUE is responsible for getting the arbiter output and sends the output to SPI master.

PMIFSPI_MST translates PMIF output information to MediaTek SPI interface protocol. It has two operation modes. One is auto mode, which works together with PMIF; the other is manual mode, in which SPI formats can be generated manually by controlling registers.

SPICIPHER is an encrypt module for encryption SPI bus. When cipher is enabled, SPI data is encrypted before being sent to the pad. PMIC wills use a decipher for data recovery.

All PMICSPI_MST control register operates in the secure domain. PMIFSPMI_MST function translates the PMIF output information using SPMI interface protocol.

3.13.2.5 Signal Descriptions

Table 3-214 presents PWRAP and SPMI signal descriptions.

Table 3-214 PWRAP Signal Descriptions

Signal Name	Type	Description	Ball Location
PWRAP SPI0			
PWRAP_SPI0_CK	DO	PWRAP PMIC SPI clock	AT21
PWRAP_SPI0_CSN	DO	PWRAP SPI chip select	AU21
PWRAP_SPI0_MI	DI	PWRAP SPI master input	AP21, AN21
PWRAP_SPI0_MO	DO	PWRAP SPI master output	AP21, AN21
PMIC SPMI			
SPMI_M_SCL	DIO	PMIC SPMI clock	AN20
SPMI_M_SDA	DIO	PMIC SPMI data	AM20

3.13.2.6 Theory of Operations

PWRAP SPI is a MediaTek-inside transmit format. It only supports MediaTek-inside PMIC peripherals. Figure 3-262 and Figure 3-263 show the transmit format.

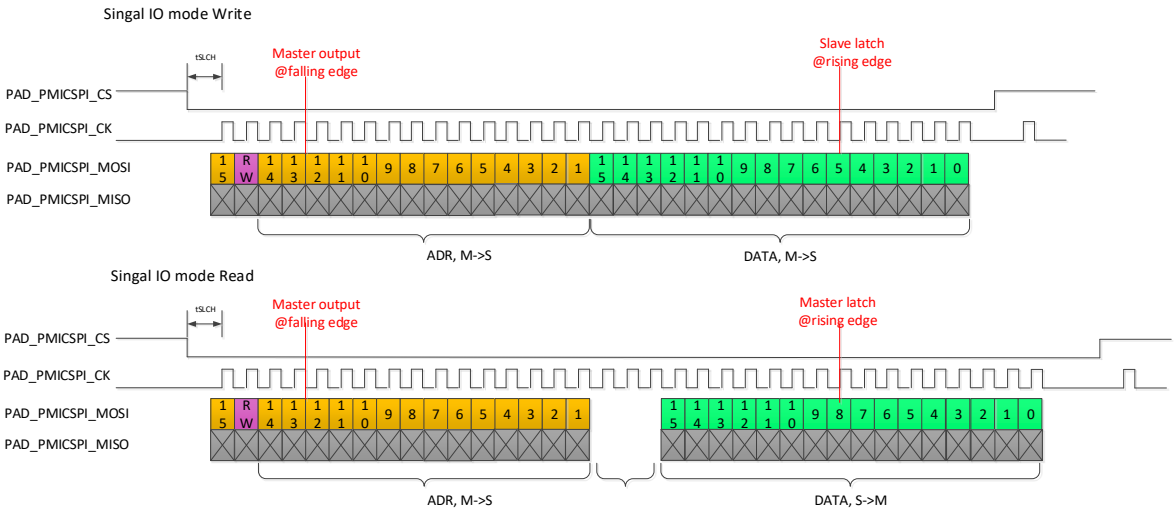


Figure 3-262 MTK_PMIC Single I/O Transaction Format

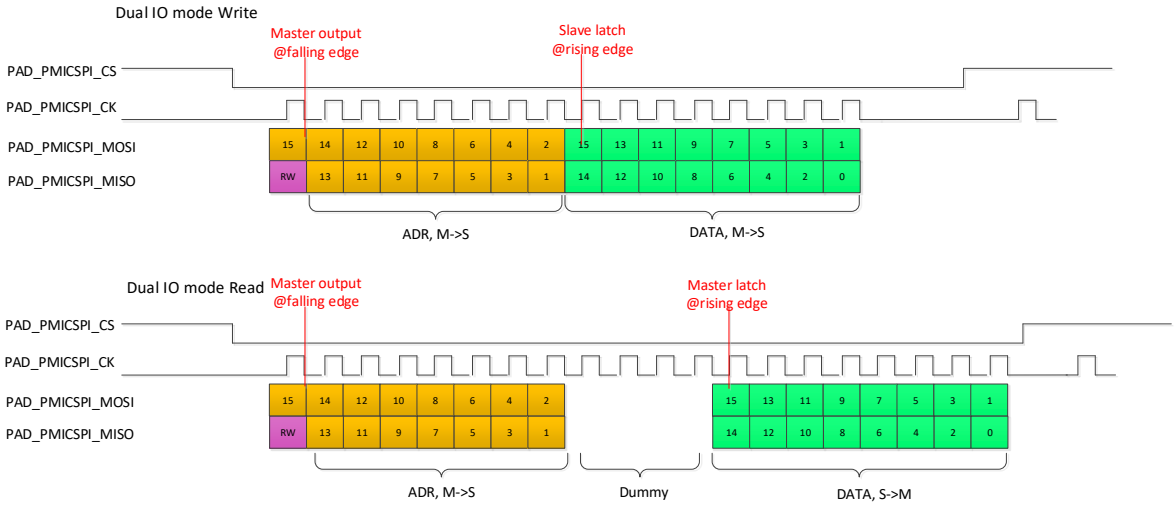


Figure 3-263 MTK_PMIC Dual I/O Transaction Format

The SPMI is a two-wire serial interface that connects the integrated Power Control (PC) of a SoC processer system with one or more PMIC voltage regulation systems. Figure 3-264 shows an example SPMI system.

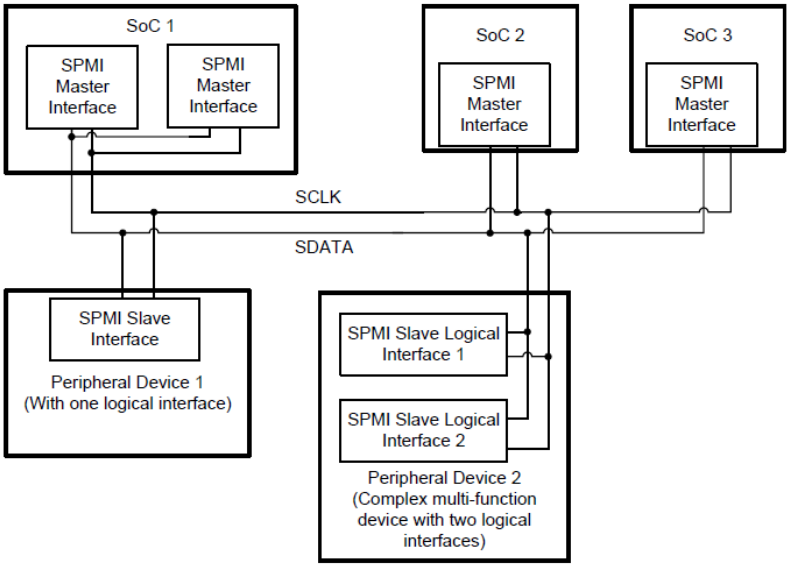


Figure 3-264 SPMI System Example

3.13.2.7 Programming Guide

Table 3-215 and Table 3-216 list the PMIF controller programming sequence.

Table 3-215 PWRAP and SPMI Controller Write Command Programming Guide

Step	Sequence	REG_Name	Comment
1	Check whether INIT_DONE is set.	SWINF_*_STA	
2	Wait for FSM state to be IDLE.	SWINF_*_STA	
3	Set the write data.	SWINF_*_WDATA_31_0	
4	Send the command.	SWINF_*_ACC	[31:30]: Command (CMD) [29]: Read/Write (WRITE)

Step	Sequence	REG_Name	Comment
			[27:24]: Slave ID (SLVID) [19:16]: Data Byte Count (BYTECNT) [15:0]: Address (ADDR)

Table 3-216 PWRAP and SPMI Controller Read Command Programming Guide

Step	Sequence	REG_Name	Comment
1	Check whether INIT_DONE is set.	SWINF_*_STA	
2	Wait for FSM state to be IDLE.	SWINF_*_STA	
3	Send the command.	SWINF_*_ACC	[31:30]: Command (CMD) [29]: Read/Write (WRITE) [27:24]: Slave ID (SLVID) [19:16]: Data Byte Count (BYTECNT) [15:0]: Address (ADDR)
4	Wait for FSM state to be WFVLDCLR.	SWINF_*_STA	
5	Get the read data.	SWINF_*_RDATA_31_0	
6	Clear the valid flag.	SWINF_*_STA	

3.13.2.8 Register Definition

Refer to "*MT8395 Register Map*" for detailed register descriptions.

3.13.3 Auxiliary Analog-to-Digital Converter (AUXADC)

3.13.3.1 Overview

The Auxiliary Analog/Digital Converter (AUXADC) module is designed to identify the plugged peripheral and perform temperature and voltage measurement. There are 16 input channels that allow diverse applications, such as temperature/voltage measurement and light sensing. The device features one AUXADC module.

3.13.3.2 Features

The module contains:

- Immediate analog-digital conversion
- Background detection and interrupt

3.13.3.3 Block Diagram

The software controls AUXADC through the APB. Once the hardware receives the command, it triggers AUXADC's channel sampling automatically. The software polls the status register or waits for interrupts from the CPU.

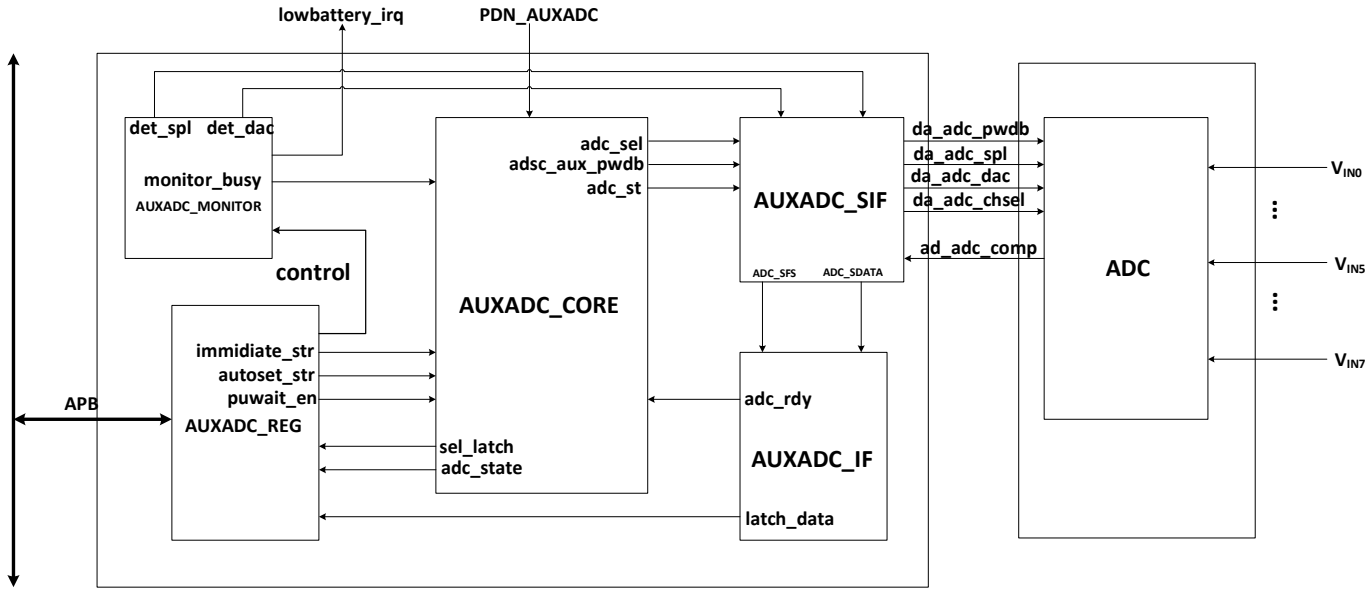


Figure 3-265 Block Diagram of AUXADC

3.13.3.4 Function Description

- Immediate analog-digital conversion
  In the immediate mode, AUXADC samples the value once only when the flag in the AUXADC_CON1 register is set. For example, if the flag IMM0 in AUXADC_CON1 is set, the module samples the data for channel 0. The IMM flags have to be cleared and set again to initialize another sampling. The value sampled for channel 0 is stored in the register AUXADC_DAT0. If the AUTOSET(x) flag in the register AUXADC_CON0 is set, the auto-sampling function is enabled in channel(x). The module samples the data for channel(x) whenever the corresponding data register is read.
  If multiple channels are selected at the same time, the tasks are performed sequentially on every selected channel from channel 15 to channel 0.
- Background detection and interrupt
  If background detection is enabled, AUXADC automatically compares the selected channel data with the user-defined value. If the results are continuously greater or less than the given value, AUXADC issues an interrupt to inform the system user.

3.13.3.5 AUXADC Signal Descriptions

Table 3-217 presents the AUXADC signal descriptions.

Table 3-217 AUXADC Signal Descriptions

Signal Name	Type	Description	Ball Location
AUXIF_CLK0	DO	Auxiliary analog/digital converter clock 0	AP16
AUXIF_CLK1	DO	Auxiliary analog/digital converter clock 1	AM16
AUXIF_ST0	DO	Auxiliary analog/digital converter status 0	AR16
AUXIF_ST1	DO	Auxiliary analog/digital converter status 1	AN16
AUXIN0[(1)(2)]	AIO	AUXADC external input channel 0	AP5
AUXIN1[(1)(2)]	AIO	AUXADC external input channel 1	AP6

Signal Name	Type	Description	Ball Location
AUXIN2[1][2]	AIO	AUXADC external input channel 2	AN6
AUXIN3[1][2]	AIO	AUXADC external input channel 3	AM6
AUXIN4[1][2]	AIO	AUXADC external input channel 4	AL6
AUXIN5[1][2]	AIO	AUXADC external input channel 5	AK6
REFP[3]	AIO	Positive reference port for internal circuit	AM7

1. This pin should be connected to GND when unused.
2. All AUXIN* pins should be connected via a 0.1-µF capacitor to GND, as close as possible to the device, when used.
3. The REFP pin should be connected via a 1-µF capacitor to GND, as close as possible to the device, when used.

3.13.3.6 AUXADC Timing and Functional Characteristics

Table 3-218 presents timing and functional characteristics for AUXADC in the device.

Table 3-218 AUXADC Specifications

Parameter		Min	Typ	Max	Unit
f_{OP}	Operating frequency		3.25		MHz
N	Resolution			12	Bit
f_S	Sampling rate at N-bit		3.25 / (N+8)		MSPS
IN_{SW}	Input swing	0.05		1.45	V
C_{IN}	Input capacitance unselected channel		50		pF
	Input capacitance selected channel		4		pF
R_{IN}	Input resistance unselected channel	400			MΩ
$F_{cycle\_latency}$	Cycle latency		N+8		$1/f_{OP}$
DNL	Differential non-linearity		+1.0/-1.0		LSB
INL	Integral non-linearity		+2.0/-2.0		LSB
SNR+D	Signal to noise and distortion ratio (1 kHz full swing input; 1.0833 MHz clock rate)	60	67		dB

3.13.3.7 Theory of Operations

Successive-Approximation Register (SAR) ADC provides low power consumption, cost-effective and medium resolution. The AUXADC module has the SAR ADC architecture.

The following shows an example of 12-bit conversion. V_{REF} is the reference voltage of AUXADC.

AUXADC implements a binary search algorithm. An initial register V_{DA} value, the mid-value between (2^{12}-1) and 0, is compared with the input voltage V_{IN}. The value represents V_{REF} /2. If V_{IN} is bigger than V_{DA}, the output of comparison is 1, and the MSB is 1. Otherwise, the MSB is 0. Subsequently, bit 11 is set to 1, and another comparison is done. Bit10 to bit 0 are executed as the previous action. Then, the 12-bit digital value is available.

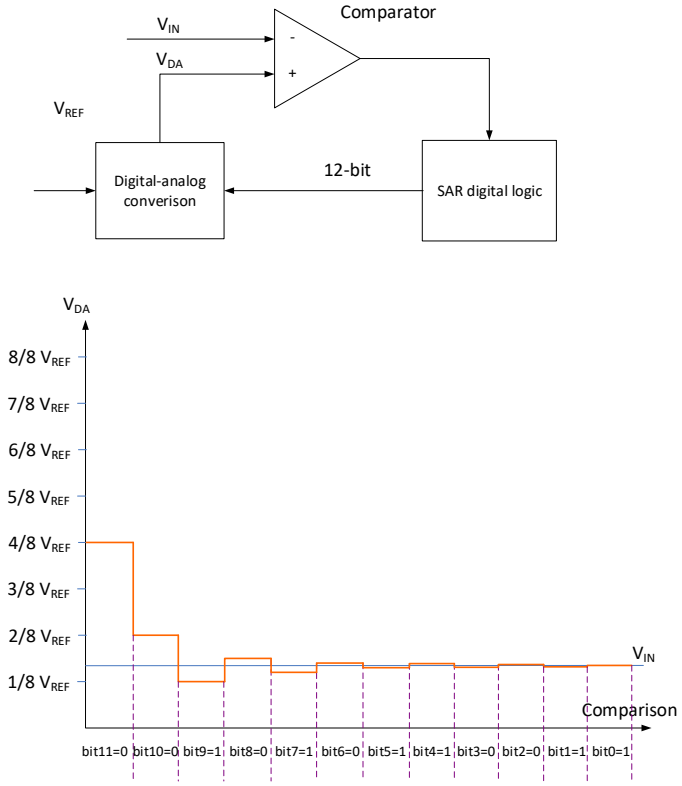


Figure 3-266 Theory of AUXADC Operation

3.13.3.8 Programming Guide

3.13.3.8.1 Immediate Mode

Table 3-219 Immediate Mode Programming Sequence

Immediate Mode						
Description	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
Open AUXADC Clock	W	INFRACFG_AO_BASE + 0x0088	[10]	AUXADC_CG_SET	1'b1	Set 0 to enable AUXADC clock.
Set Immediate Mode	W	AUXADC_BASE + 0x0004	[15:0]	AUXADC_CON1	USER_DEFINED	Set 1 to sample corresponding channel once.
Polling Ready	R	AUXADC_BASE + 0x0014 + n*4	[12]	RDYn	1'b1	Sample data is ready when this bit changes to 1.
Read Result	R	AUXADC_BASE + 0x0014 + n*4	[11:0]	DATn	-	Sample result

3.13.3.8.2 Background Detection

Table 3-220 Background Detection Programming Sequence

Background Detection						
Description	**R/W**	**Address**	**Bit**	**MACRO**	**Value**	**Note**
Set Threshold Voltage	W	AUXADC_BASE + 0x0084	[11:0]	VOL	USER_DEFINED	Set Threshold Voltage
Set Compare direction	W	AUXADC_BASE + 0x0084	[12]	INV	USER_DEFINED	0: Lower 1: Higher
Set Detection Channel	W	AUXADC_BASE + 0x0088	[3:0]	CHSEL	USER_DEFINED	Set channel to be sampled in background.
Set Detection Period	W	AUXADC_BASE + 0x008C	[13:0]	BG_DET_PERIOD	USER_DEFINED	Background sample period: When this value is not 0, the background detection is activated automatically and other ADC sampling functions are stopped. The counter counts by 32K clock. When counter value is greater than DET_PERIOD, the detection is activated.
Set Detection De-bounce	W	AUXADC_BASE + 0x0090	[13:0]	BG_DET_TIME	USER_DEFINED	Background de-bounce time: When the number of the detected channel is higher or lower than the pre-defined voltage and exceeds "debounce_time", the interrupt is issued.

3.13.4 Thermal Control Subsystem (TCSYS)

3.13.4.1 Introduction

Thermal management is crucial on the SoC platform. Through thermal management, an SoC can operate within specific temperature constraints whole fulfilling computing performance requirements. Operations under over-temperature condition for a long period of time may cause reliability issues.

The thermal management system includes several thermal sensors embedded in possible hotspots on the die and a thermal controller module for periodic measurement for each hotspot. The measurement results are read by the software. However, in order to minimize software efforts for monitoring temperature, the thermal controller generates interrupts to a system handler for abnormal conditions.

3.13.4.2 Features

The thermal management system includes the following features:

- Support up to four-thermal sensors per hardware module unit.
- Programmable periodic temperature measurement.

- Two independent Finite State Machines (FSMs) for temperature monitoring.
- Different types of low pass filters for thermal sensor readings.

3.13.4.3 Block Diagram

There are three major building blocks for the thermal control subsystem, as Figure 3-267 shows.

- Sensing device: Thermal Sensing Micro-Circuit Unit (TSMCU)
- Convertor: Low Voltage Thermal Sensor (LVTS) convertor
- Digital controller: LVTS_CTRL

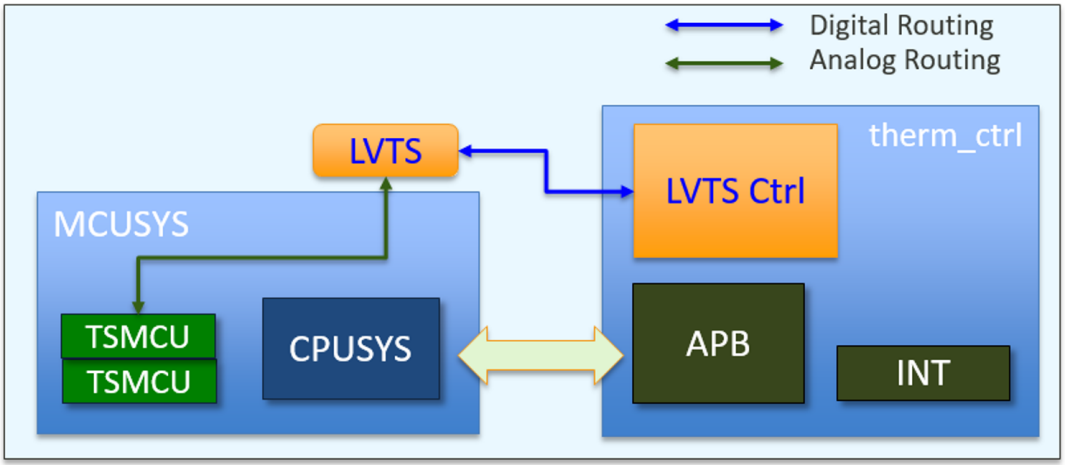


Figure 3-267 Thermal Control Subsystem

3.13.4.4 Function Description

Thermal controller periodically polls all sensors for SoC operating within a pre-defined temperature range to avoid function failure and reliability issues. According to the temperature measurement, the system performance can be adjusted for a system design with power dissipation being monitored. Figure 3-268 shows the temperature measurement scheme. Note that the hottest location in an SoC may vary in different applications.

When the thermal controller informs the software of an abnormal condition, the consecutive power reduction methodology should be efficient and with low latency.

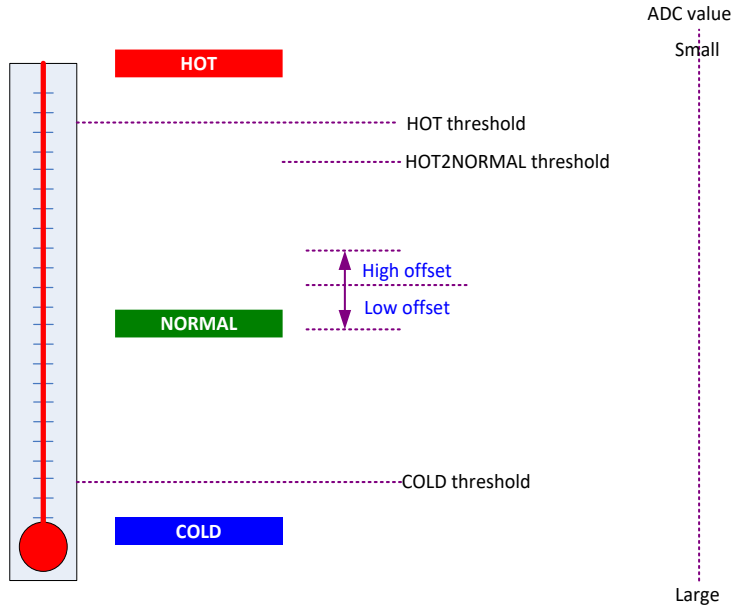


Figure 3-268 Block Diagram of System Temperature Measurement Scheme

3.13.4.5 Theory of Operations

3.13.4.5.1 Interrupt Control

Figure 3-269 shows the interrupt conditions of high and low temperature monitors. Software can determine which temperature sensors to be monitored. Once any of the following three interrupt conditions occur in any of the monitored temperature sensors, an interrupt is generated.

- Cold interrupt: Asserted when the temperature crosses and falls below the cold threshold.
- Hot interrupt: Asserted when the temperature crosses and rises above the hot threshold.
- Hot-to-normal interrupt: Asserted when the temperature crosses and falls below the hot-to-normal threshold.

Figure 3-270 shows the Finite State Machine (FSM) diagram. The states are as listed below

- COLD_ST: The temperature is lower than the cold threshold.
- NORMAL_ST: The temperature is within the hot-to-normal threshold and cold threshold.
- HOT1_ST: The temperature is within the hot-to-normal threshold and hot threshold, and the previous state is NORMAL_ST.
- VERY_HOT_ST: The temperature is higher than the hot threshold.
- HOT2_ST: The temperature is within the hot-to-normal threshold and hot threshold, and the previous state is VERY_HOT_ST.

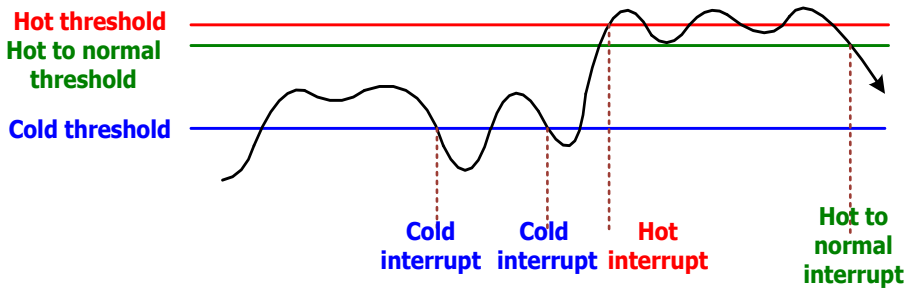


Figure 3-269 Interrupt Conditions for High/Low Temperature Monitoring

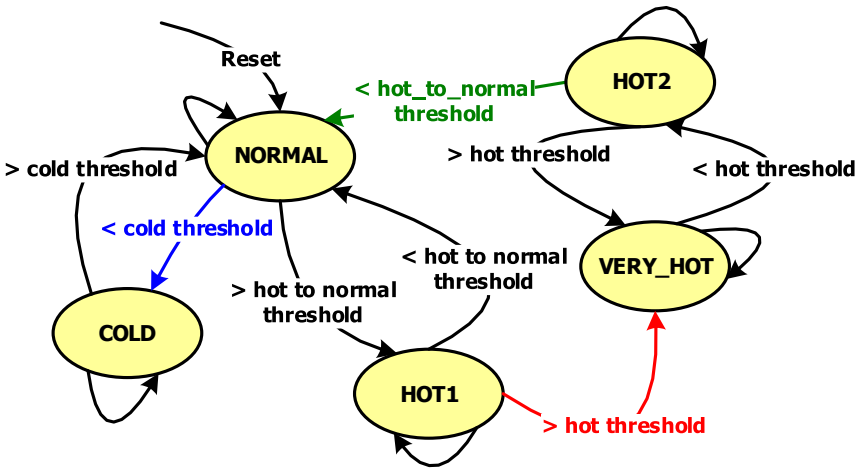


Figure 3-270 Finite State Machine for High/Low Temperature Monitoring

The system also provides a feature to control the junction temperature within the pre-defined HIGH-OFFSET and LOW-OFFSET. Figure 3-271 depicts the concept of the feature and Figure 3-272 shows the FSM. There are two interrupts and two alarms:

- LOW-OFFSET alarm to RGU: Asserted when the temperature crosses and drops below the low offset.
- HIGH-OFFSET alarm to RGU: Asserted when the temperature crosses and rises above the high offset.
- LOW-OFFSET Interrupt: Asserted when the temperature crosses and drops below the low offset. The FSM transitions from the NORMAL state into the LOW OFFSET state.
- HIGH-OFFSET Interrupt: Asserted when the temperature crosses and rises above the high offset. The FSM transitions from the NORMAL state into the HIGH OFFSET state.

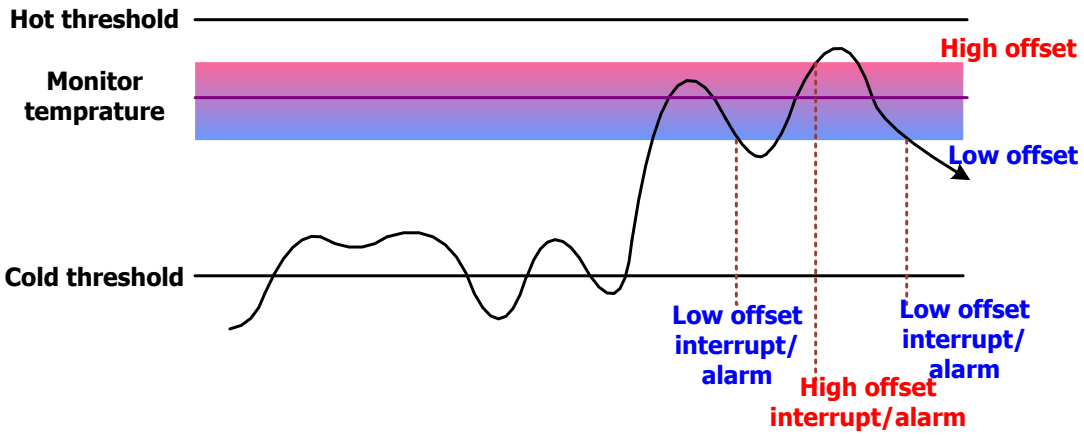


Figure 3-271 Interrupt Conditions of High/Low OFFSET Monitoring

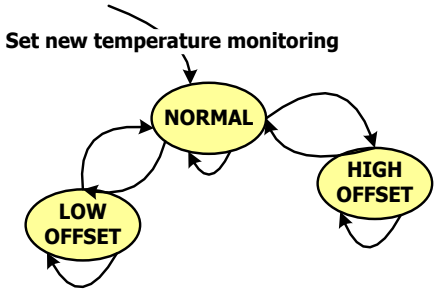


Figure 3-272 Finite State Machine for High/Low OFFSET Monitoring

3.13.4.5.2 Thermal Sensor Locations

Thermal sensors are placed at different locations within a die for monitoring the junction temperatures of the different die locations.

Table 3-221 lists the locations of all thermal sensors.

Table 3-221 Locations Monitored by Temperature Sensors

Core	Module	Sensor
#1	BIG-CPU (Bottom)	TS-1 and TS-2
#2	BIG-CPU (Top)	TS-3 and TS-4
#3	CPU	TS-5 to TS-8
#4	APU	TS-9 and TS-10
#5	GPU	TS-11 and TS-12
#6	SOC/TOP	TS-13 to TS-15
#7	CAM	TS-16 and TS-17

Note:

- The difference between BIG-CPU and CPU is that CPU represents cores other than Big Cores

3.13.4.6 Programming Guide

Table 3-222 lists the thermal controller programming sequence.
TMU_BASE (CPU View):

- THERM_CTRL_AP register base address: 0x1100B000
- THERM_CTRL_MCU register base address: 0x11278000

Table 3-222 Thermal Controller Programming Guide

Step	Sequence	REG_Name	REG_Value	Address offset
1	Enable controller clock	LVTSCLKEN	0x00000001	TMU_BASE+0x00E4
2	Establish LVTS connection (Set in order, 2 us delay for each command)	LVTS_CONFIG	0xC103FFFF 0xC502FF55	TMU_BASE+0x0050
3	Check LVTS connection ID	LVTS_ID	Read Only	TMU_BASE+0x004C
4	Initialize LVTS device	LVTS_CONFIG	0xC1030E01	TMU_BASE+0x0050

Step	Sequence	REG_Name	REG_Value	Address offset
	(Set in order, 2 us delay for each command)		0xC1030CFC 0xC1030A8C 0xC103098D 0xC10308F1 0xC10307A6 0xC10306b8 0xC1030500 0xC1030420 0xC1030300	
5	Prepare calibration data	LVTSEDATA00 LVTSEDATA01 LVTSEDATA02 LVTSEDATA03	0x00xxxxxx 0x00xxxxxx 0x00xxxxxx 0x00xxxxxx	TMU_BASE+0x0054 TMU_BASE+0x0058 TMU_BASE+0x005c TMU_BASE+0x0060
6	Configure PNP data to select the monitored sensors	LVTSTSSEL	0x13121110	TMU_BASE+0x0040
7	Configure sampling method for the thermal sensors (optional)	LVTSMSRCTL0	0x000006DB	TMU_BASE+0x0038
8	Configure measurement period and cycles (optional)	LVTSMONCTL1 LVTSMONCTL2	0x0000000C 0x000101AD	TMU_BASE+0x0004 TMU_BASE+0x0008
9	Trigger events setting (optional)	LVTSMONINT	User-defined	TMU_BASE+0x000C
10	Enable periodical measurement for the thermal sensors	LVTSMONCTL0	0x0000020F	TMU_BASE+0x0000
11	Read thermal sensors measurement results	LVTSMSR0 LVTSMSR1 LVTSMSR2 LVTSMSR3	Read only	TMU_BASE+0x0090 TMU_BASE+0x0094 TMU_BASE+0x0098 TMU_BASE+0x00B8

3.14 Boot Flash

The boot sequence is eMMC->UFS->SPI NOR. When a boot device (e.g. eMMC) fails to load bootloader, the bootrom will let the next boot device (e.g. UFS) try to load the bootloader. If users want to skip a specific boot device, a corresponding eFuse can be used to disable it.

The device supports the following boot flash:

- eMMC
- UFS
- SPI NOR

3.15 ROM Power Down Mode

After system boot, ROM can be powered down and prevented from any probe of ROM content.

4 Ball Map

Figure 4-1 presents simplified diagram of the location of the balls on the package.

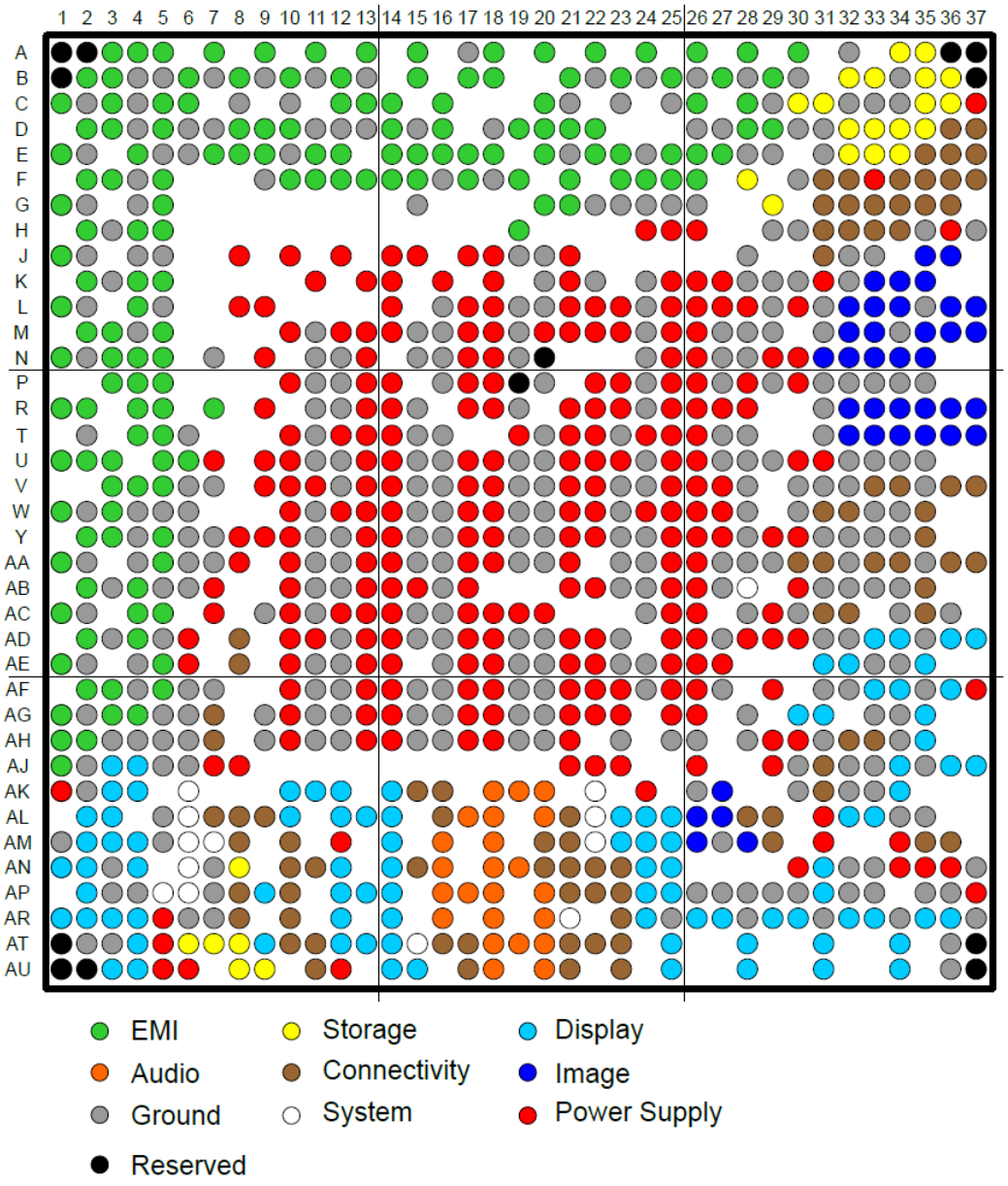


Figure 4-1 Ball Map Diagram

For detailed information about package outlines, thermal characteristics, and markings, see Section 7 Package Information.

4.1 Quadrant Pinout

Figure 4-2 shows a top view mapping of the package quadrants.

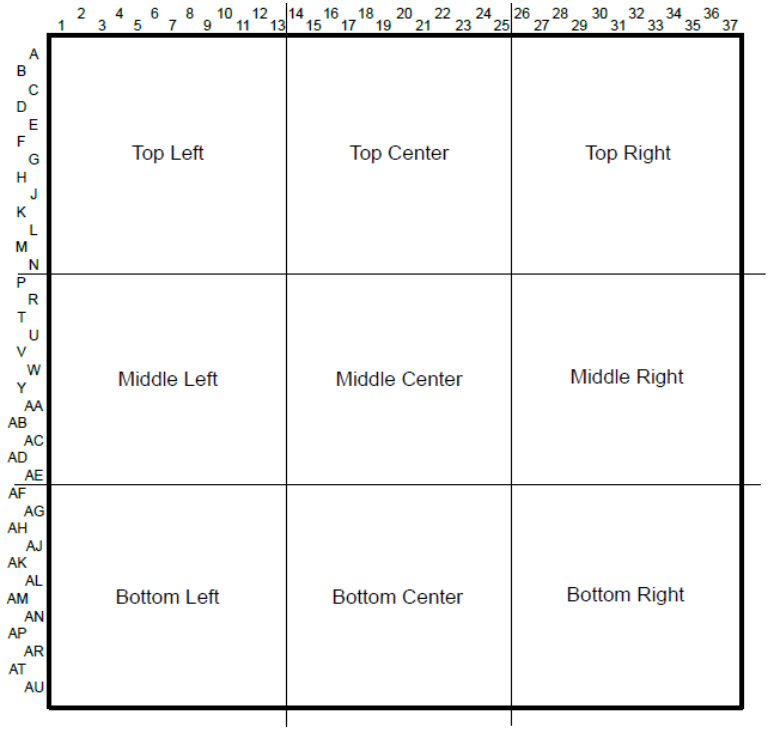


Figure 4-2 Package Quadrants Mapping

Table 4-1 shows pin mapping on the top left part of the package.

Table 4-1 Ball Map—Top Left

	1	2	3	4	5	6	7	8	9	10	11	12	13
A	DUMMY	DUMMY	EMI1_DQ2	EMI1_DQ5	EMI1_DMI0		EMI1_DQ7		EMI1_DQ8		EMI1_DMI1		EMI1_CA4
B	DUMMY	EMI2_RESET_N	EMI0_EXTR	DVSS	DVSS	EMI1_DQ6	DVSS	EMI1_DQ15	DVSS	EMI1_DQ9	DVSS	EMI1_DQ14	DVSS
C	EMI2_DQ0	DVSS	EMI2_DQS0_T	DVSS	EMI1_DQS0_T	EMI1_DQ0		DVSS		DVSS		EMI1_DQ13	EMI1_CA3
D		EMI2_DQ1	EMI2_DQS0_C	DVSS	EMI1_DQS0_C	DVSS	DVSS	EMI1_DQ3	EMI1_DQS1_T	EMI1_DQ10	DVSS	DVSS	DVSS
E	EMI2_DMI0	DVSS		EMI2_DQ5	DVSS	DVSS	EMI1_DQ1	EMI1_DQ4	EMI1_DQS1_C	DVSS	NC-PAD_BRCKE1	EMI1_DQ12	
F		EMI2_DQ6	EMI2_DQ2	DVSS	EMI2_DQ4				DVSS	EMI1_DQ11	NC-PAD_BRCKE0	NC-PAD_BRCS1	NC_PAD_BRDQ7_B1
G	EMI2_DQ7	DVSS		DVSS	EMI2_DQ3								
H		EMI2_DQ15	DVSS	EMI2_DQS1_T	EMI2_DQS1_C								
J	EMI2_DQ8	DVSS		DVSS	DVSS			AVDD18_EMI		AVDD075_EMI0		AVDD075_EMI0	
K		EMI2_DQ9	DVSS	EMI2_DQ11	EMI2_DQ10						AVDDQ_EMI0		AVDDQ_EMI0
L	EMI2_DMI1	DVSS		EMI2_DQ13	DVSS			AVDD12_EMI	AVDD075_EMI2				
M		EMI2_DQ14	EMI2_DQ12	DVSS	EMI2_CS0					AVDDQ_EMI2	DVSS	DVDD_CORE	DVDD_CORE
N	EMI2_CA4	DVSS	EMI2_CA3	EMI2_CS1	EMI2_CA0		DVSS		AVDD075_EMI2		DVSS	DVSS	DVDD_SRAM_CORE

Table 4-2 shows pin mapping on the top center part of the package.

Table 4-2 Ball Map—Top Center

	14	15	16	17	18	19	20	21	22	23	24	25
A		EMI1_CA2		DVSS	EMI0_CA2		EMI0_CA4		EMI0_DMI1		EMI0_DQ8	
B		EMI1_CA5		EMI0_CA1	EMI0_CA0			EMI0_DQ14	DVSS	EMI0_DQ9	DVSS	EMI0_DQ15
C	EMI1_CKE1		EMI1_CA1				EMI0_CA3	DVSS		DVSS		DVSS
D	EMI1_CKE0	DVSS	EMI1_CK_C		DVSS	EMI0_CA5	NC-PAD_ ARDQ7_B1	EMI0_DQ12	EMI0_DQ13			
E	NC-PAD_ BRDQ6_B1	EMI1_CA0	EMI1_CK_T	EMI0_CK_T	EMI0_CS1		NC-PAD_ ARDQ6_B1	DVSS	NC-PAD_ ARCKE0	EMI0_DQ11	DVSS	EMI0_ DQS1_T
F	EMI1_CS0	EMI1_CS1	DVSS	EMI0_CK_C	DVSS	EMI0_CS0		NC-PAD_ ARCS1		NC-PAD_ ARCKE1	EMI0_DQ10	EMI0_ DQS1_C
G		DVSS					EMI0_CKE1	EMI0_CKE0	DVSS	DVSS	DVSS	DVSS
H						EMI0_TP					AVDD12_ UFS	AVDD12_ CKBUF_UFS
J	AVDD075_ EMI0	AVDD075_ EMI0		AVDD075_ EMI0	AVDD075_ EMI0	DVSS	DVSS	DVDD_CORE				
K	AVDDQ_ EMI0		AVDDQ_ EMI0		AVDDQ_ EMI0		DVSS	DVDD_CORE	DVSS		DVSS	DVDD_ PROC_B
L	DVDD_CORE		DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVDD_CORE	DVSS	DVDD_ PROC_B
M	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_ SRAM_CORE	DVSS	DVDD_CORE	DVDD_CORE	DVDD_ SRAM_CORE	DVDD_CORE	DVSS	DVDD_ PROC_B
N		DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	TN_APPLLGP				DVSS	DVDD_ PROC_B

Table 4-3 shows pin mapping on the top right part of the package.

Table 4-3 Ball Map—Top Right

26	27	28	29	30	31	32	33	34	35	36	37	
EMI0_DQ7		EMI0_DMI0		EMI0_DQ2		DVSS		UFS_RST_N	EMMC_DSL	DUMMY	DUMMY	A
DVSS	EMI0_DQ6	DVSS	EMI0_DQ5	DVSS		UFS_RX0N	UFS_RX0P	DVSS	EMMC_RSTB	EMMC_CLK	DUMMY	B
EMI0_DQ4		EMI0_DQS0_T	DVSS	UFS_TX0P	UFS_TX0N	DVSS	DVSS	DVSS	EMMC_DAT3	EMMC_CMD	DVDD18_IOEMMC	C
DVSS	DVSS	EMI0_DQS0_C	EMI0_RESET_N	DVSS	DVSS	EMMC_DAT2	EMMC_DAT1	EMMC_DAT0	EMMC_DAT4	SDA0	SCL0	D
EMI0_DQ3	EMI0_DQ0	DVSS	DVSS		DVSS	EMMC_DAT7	EMMC_DAT6	EMMC_DAT5	SCL1	SDA1	SDA2	E
EMI0_DQ1		UFS_REFCK_OUT		DVSS	SPIM0_CSB	SPIM0_CLK	DVDD18_IOT	SCL3	SDA3	SDA4	SCL2	F
DVSS			UFS_PLL_CKREF		SPIM0_MI	SPIM0_MO	GPIO_03	GPIO_01	GPIO_04	SCL4		G
AVDD18_UFS			DVSS	DVSS	GPIO_00	GPIO_06	GPIO_05	GPIO_07	DVSS	DVDD28_MSDC2	DVSS	H
		DVSS			GPIO_02	DVSS	DVSS		CSI0A_L0P	CSI0A_L0N		J
DVDD_PROC_B	DVDD_PROC_B	DVSS	DVSS	DVSS	DVDD18_MSDC2	DVSS	CSI0A_L2N	CSI0A_L2P	CSI0A_L1P			K
DVDD_PROC_B	DVDD_PROC_B	DVDD_PROC_B	DVSS	DVDD_CORE	DVSS	CSI0B_L0P	CSI0B_L0N	CSI0A_L1N	DVSS	CSI0D_L0P	CSI0D_L0N	L
DVDD_PROC_B	DVSS	DVSS	DVSS		DVSS	CSI0B_L1N	CSI0B_L1P	DVSS	CSI0C_L0N	CSI0D_L1N	CSI0D_L1P	M
DVDD_PROC_B	DVSS	DVSS	DVDD_SRAM_PROC_B	AVDD18_CSI	CSI0C_L2P	CSI0C_L2N	CSI0C_L1N	CSI0C_L1P	CSI0C_L0P			N

Table 4-4 shows pin mapping on the middle left part of the package.

Table 4-4 Ball Map—Middle Left

	1	2	3	4	5	6	7	8	9	10	11	12	13
P			EMI2_CA5	EMI2_CA2	EMI2_CA1					AVDDQ_EMI2	DVSS	DVSS	DVDD_CORE
R	EMI2_CKE1	EMI2_CKE0		EMI2_CK_T	EMI2_CK_C		EMI2_TP		AVDD075_EMI2		DVSS	DVSS	DVDD_CORE
T		DVSS		EMI3_CK_T	EMI3_CK_C	DVSS				AVDDQ_EMI2	DVSS	DVDD_SRAM_CORE	DVDD_CORE
U	EMI3_CKE1	EMI3_CKE0	EMI3_CA2		EMI3_CA1	EMI3_CA0	DVSS		AVDD075_EMI2	AVDDQ_EMI2	DVSS	DVSS	DVDD_CORE
V			EMI3_CA5	EMI3_CS1	EMI3_CS0	DVSS	DVSS		AVDD075_EMI2	AVDDQ_EMI2	AVDD12_APPLLGP2	DVSS	DVDD_CORE
W	EMI3_CA4	DVSS	EMI3_CA3	DVSS	DVSS	DVSS				AVDDQ_EMI2	DVSS	AVDD18_APPLLGP2	DVDD_CORE
Y		EMI3_DQ14	EMI3_DQ13	DVSS	EMI3_DQ12	DVSS	DVSS	AVDD18_USB_P2	AVDD075_EMI2	DVDD_GPU	DVSS	DVSS	DVDD_GPU
AA	EMI3_DMI1	DVSS		DVSS	EMI3_DQ11	DVSS	DVSS	AVDD18_USB_P3		DVDD_GPU	DVSS	DVSS	DVDD_GPU
AB		EMI3_DQ9	DVSS	EMI3_DQ10	DVSS	DVSS	AVDD33_USB_P2			DVDD_GPU	DVSS	DVSS	DVDD_GPU
AC	EMI3_DQ8	DVSS		EMI3_DQS1_T	EMI3_DQS1_C		AVDD33_USB_P3		DVSS	DVDD_GPU	DVSS	DVDD_SRAM_GPU	DVDD_GPU
AD		EMI3_DQ15	DVSS	EMI3_DQ3	DVSS	AVDD12_USB_P2		USB_DP_P2		DVDD_GPU	DVDD_SRAM_GPU	DVSS	DVDD_GPU
AE	EMI3_DQ7	DVSS		DVSS	EMI3_DQ4	AVDD12_USB_P3		USB_DM_P2		DVDD_GPU	DVSS	DVSS	DVDD_GPU

Table 4-5 shows pin mapping on the middle center part of the package.

Table 4-5 Ball Map—Middle Center

	14	15	16	17	18	19	20	21	22	23	24	25
P	DVDD_CORE		DVSS	DVDD_CORE	DVDD_CORE	TP_APPLLGP	DVSS		AVDD12_APPLLGP1	DVDD_CORE	DVSS	DVDD_PROC_L
R	DVDD_CORE	DVSS		DVDD_CORE	DVDD_CORE	DVSS		AVDD18_APPLLGP1	DVDD_CORE	DVDD_CORE	DVSS	DVDD_PROC_L
T	DVDD_CORE	DVSS	DVSS			DVDD_SRAM_CORE	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVDD_PROC_L	DVDD_PROC_L
U	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVDD_SRAM_PROC_B	DVSS	DVDD_PROC_B
V	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_PROC_B
W	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVDD_PROC_B	DVDD_PROC_B
Y	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_PROC_B
AA	DVDD_GPU	DVSS	DVSS	DVDD_SRAM_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE		DVSS	DVSS	DVDD_CORE
AB	DVDD_GPU	DVDD18_VQPS	DVSS	DVDD_DLA				DVDD_CORE	DVDD_CORE	DVSS	DVSS	DVDD_CORE
AC	AVDD12_APPLLGP4	DVSS	DVSS	DVDD_DLA	DVDD_DLA	AVDD18_APPLLGP_APU	AVDD12_APPLLGP_APU				DVSS	DVDD_CORE
AD	AVDD18_APPLLGP4	DVSS	DVSS	DVDD_DLA	DVDD_DLA	DVSS	DVSS	DVDD_APU	DVDD_APU	DVSS		DVDD_CORE
AE	DVDD_GPU		DVSS	DVDD_DLA	DVDD_SRAM_APU	DVSS	DVSS	DVDD_APU	DVDD_APU	DVSS	DVSS	DVDD_SRAM_CORE

Table 4-6 shows pin mapping on the middle right part of the package.

Table 4-6 Ball Map-Middle Right

26	27	28	29	30	31	32	33	34	35	36	37	
DVDD_PROC_L	DVSS	AVDD18_PROC	DVSS	AVDD12_CSI	DVSS	DVSS	DVSS	DVSS	DVSS			P
DVDD_PROC_L	DVDD_PROC_L	DVDD_SRAM_PROC_L			DVSS	CSI1A_L0P_T0A	CSI1A_L0N_T0B	CSI1A_L1P_T0C	CSI1A_L1N_T1A	CSI1A_L2P_T1B	CSI1A_L2N_T1C	R
DVDD_PROC_L	DVSS	DVSS			DVSS	CSI1B_L2N_T1C	CSI1B_L2P_T1B	CSI1B_L1N_T1A	CSI1B_L1P_T0C	CSI1B_L0P_T0A	CSI1B_L0N_T0B	T
DVDD_PROC_B	DVSS	DVSS	DVSS	AVDD18_CKSQ	AVDD12_CKSQ	DVSS	DVSS	DVSS	DVSS			U
DVDD_PROC_B	DVDD_PROC_B	DVSS		DVSS	DVSS	DVSS	PCIE_LN0_RXP_P1	PCIE_LN0_RXN_P1	DVSS	PCIE_CKRXP_P1	PCIE_CKRXN_P1	V
DVDD_PROC_B	DVDD_PROC_B	DVSS		DVSS	PCIE_LN0_TXP_P1	PCIE_LN0_TXN_P1	DVSS	DVSS	PCIEG3_LN0_RXN			W
DVDD_PROC_B	DVDD_PROC_B	DVSS	AVDD12_PCIE_P1	AVDD18_PCIE_P1	DVSS	DVSS	DVSS	DVSS	PCIEG3_LN0_RXP			Y
DVDD_CORE	DVSS	DVSS	DVSS	PCIEG3_LN0_TXP	PCIEG3_LN0_TXN	DVSS	PCIEG3_ CLKN	PCIEG3_ CLKP	DVSS	PCIEG3_LN1_TXN	PCIEG3_LN1_TXP	AA
DVDD_CORE	DVSS	X26M_IN		AVDD18_PCIEG3	DVSS	DVSS	DVSS	DVSS	PCIEG3_LN1_RXN			AB
DVDD_CORE		DVSS	AVDD12_PCIEG3	DVSS	USB_DP_P1	USB_DM_P1		DVSS	PCIEG3_LN1_RXP	DVSS		AC
DVDD_CORE	DVSS	AVDD12_DPTX	AVDD18_USB_P1	AVDD33_USB_P1	DVSS	DVSS	EDP_LN0_TXN	EDP_LN0_TXP	DVSS	EDP_LN1_TXP	EDP_LN1_TXN	AD
DVDD_CORE	DVDD_CORE				EDPAUXN	EDPAUXP	DVSS	DVSS	EDP_LN2_TXP			AE

Table 4-7 shows pin mapping on the bottom left part of the package.

Table 4-7 Ball Map—Bottom Left

	1	2	3	4	5	6	7	8	9	10	11	12	13
AF		EMI3_DQ6	EMI3_DQ5	DVSS	EMI3_DQ2	DVSS	DVSS			DVDD_GPU	DVSS	DVSS	DVDD_GPU
AG	EMI3_DMI0	DVSS	EMI3_ DQS0_C	EMI3_ DQS0_T	DVSS	DVSS	USB_DM_ P3		DVSS	DVDD_GPU	DVSS	DVSS	DVDD_GPU
AH	EMI3_DQ1	EMI3_DQ0	DVSS	DVSS	DVSS	DVSS	USB_DP_P3		DVSS	DVDD_GPU	DVSS	DVSS	DVDD_GPU
AJ	EMI2_EXTR	DVSS	DSI0_D2P_ T0A	DSI0_D2N_ T0B	DVSS	DVSS	DVDD18_ IOBR	DVDD_ SRAM_CORE					
AK	AVDD12_ DSI	DVSS	DSI0_D0N_ T1A	DSI0_D0P_ T0C		AUXIN5				DGI_D7	DGI_D6	DGI_D11	
AL		DSI0_CKN_ T1C	DSI0_D1P_ T2A		DVSS	AUXIN4	USB_IDDIG_ 1P	USB_DRV_ VBUS_1P	USB_IDDIG	DSI_LCM_ RST		DGI_D10	DGI_D13
AM	DVSS	DSI0_CKP_ T1B	DSI0_D1N_ T2B	DSI0_D3P_ T2C	DVSS	AUXIN3	REFP	UART1_RTS		USB_DRV_ VBUS		DVDD28_ IODGI	
AN	DSI1_D2N_ T0B	DSI1_D2P_ T0A	DVSS	DSI0_D3N		AUXIN2	DVSS	MSDC1_ CMD		KPROW1	KPROW0	DGI_D15	
AP		DSI1_D0P_ T0C	DVSS	DVSS	AUXIN0	AUXIN1	DVSS	UART1_RXD	DISP_ PWM0	UART0_RXD		DGI_D12	DGI_D14
AR	DSI1_D0N_ T1A	DSI1_CKP_ T1B	DSI1_CKN_ T1C	DSI1_D3N	AVDD18_ AUXADC	DVSS	DVSS	UART1_TXD		UART0_TXD		DGI_D9	
AT	DUMMY	DVSS	DVSS	DSI1_D3P_ T2C	AVDD12_ AUXADC	MSDC1_ DAT0	MSDC1_ DAT1	MSDC1_ DAT2	DSI_DSI_TE	UART1_CTS	KPCOL1	DGI_DE	DGI_VSYNC
AU	DUMMY	DUMMY	DSI1_D1P_ T2A	DSI1_D1N_ T2B	DVDD18_ MSDC1	DVDD28_ MSDC1		MSDC1_ DAT3	MSDC1_CLK		KPCOL0	DVDD18_ IODGI	

Table 4-8 shows pin mapping on the bottom center part of the package.

Table 4-8 Ball Map—Bottom Center

	14	15	16	17	18	19	20	21	22	23	24	25
AF	DVDD_GPU	DVSS	DVSS	DVDD_DLA	DVDD_DLA	DVSS	DVSS	DVDD_ SRAM_APU	DVDD_APU	AVDD12_ APPLLGP3	DVSS	DVDD_CORE
AG	DVDD_GPU	DVSS	DVSS	DVDD_DLA	DVDD_DLA	DVSS	DVSS	DVDD_APU	DVDD_APU	AVDD18_ APPLLGP3		DVDD_CORE
AH	DVDD_GPU	DVSS	DVSS	DVDD_DLA	DVDD_DLA	DVSS	DVSS	DVDD_APU		DVSS		DVSS
AJ								DVDD_CORE	DVDD18_ IOBM	DVDD_CORE		
AK	DGI_D3	AUD_DAT_ MISO0	AUD_SYNC_ MOSI		I2SO2_BCK	I2SO1_BCK	I2SIN_BCK		TESTMODE		AVDD18_ HDMIRX21	
AL	DGI_D2		AUD_CLK_ MOSI	DMIC1_DAT	I2SO2_MCK		I2SIN_MCK	PMIC_ SRCLKEN_IN1	SYSRSTB	HDMITX_SCL	HDMITX_ PWR5V	HDMITX_SDA
AM	DGI_D1		PCM_SYNC		I2SO1_D3		SPMI_M_ SDA	PMIC_ SRCLKEN_IN0	PMIC_ WATCHDOG	HDMITX_CEC	HDMITX_ HTPLG	HDMIRX_SCL
AN	DGI_D0	AUD_DAT_ MOSI1	PCM_DI		I2SO1_D2	I2SO1_MCK	SPMI_M_ SCL	PWRAP_ SPI_MI	SPIM2_MO	SPIM2_CSB	HDMIRX_ PWR5V	HDMIRX_ HTPLG
AP	DGI_D8		PCM_CLK	DMIC1_SCK	I2SO1_D1		I2SIN_D0	PWRAP_ SPI_MO	SPIM2_CLK	SPIM2_MI	HDMIRX_SDA	DPTX_HPD
AR	DGI_D5		PCM_DO		I2SO1_D0		I2SIN_WS	PMIC_ RTC32K_CK		SPIM1_MI	AVDD08_ HDMIRX21	DVSS
AT	DGI_CK	SCP_VREQ_ VAO	AUD_DAT_ MOSI0	AUD_DAT_ MISO2	DMIC2_DAT	I2SO1_WS	I2SO2_D0	PWRAP_ SPI_CK	SPIM1_CSB	SPIM1_MO		HDMIRX21_ CLK_P
AU	DGI_HSYNC	DGI_D4		AUD_DAT_ MISO1	DMIC2_SCK		I2SO2_WS	PWRAP_ SPI_CSN		SPIM1_CLK		HDMIRX21_ CLK_M

Table 4-9 shows pin mapping on the bottom right part of the package.

Table 4-9 Ball Map—Bottom Right

26	27	28	29	30	31	32	33	34	35	36	37	
DVDD_CORE	DVSS		AVDD18_ DPTX		DVSS	DVSS	EDP_LN3_ TXN	EDP_LN3_ TXP	DVSS	EDP_LN2_ TXN	AVDD18_ EDPTX	AF
DVDD_CORE		DVSS		DPAUXP	DPAUXN		DVSS	DVSS	DP_LN0_TXP			AG
DVSS		DVSS	AVDD12_ EDPTX	AVDD12_ USB_P1	DVSS	SSUSB_TXN	SSUSB_TXP	DVSS	DP_LN0_ TXN			AH
AVDD33_ HDMIRX21			AVDD12_ HDMITX21	DVSS	SSUSB_RXP	DVSS	DVSS	DP_LN2_TXP	DVSS	DP_LN1_TXP	DP_LN1_ TXN	AJ
DVSS	CMMCLK1			DVSS	SSUSB_RXN	DVSS	DVSS	DP_LN2_ TXN				AK
CMMRST	CMMCLK0	PCIE_ PERESET_N	PCIE_ CLKREQ_N		AVDD18_ HDMITX21	DP_LN3_TXP	DP_LN3_ TXN	DVSS	DVSS			AL
CMMPDN	DVSS	CMMCLK2	PCIE_WAKE_ N		AVDD18_ EARCRX			AVDD33_ USB_P0	USB_DP_P0	USB_DM_P0		AM
				AVDD12_ EARCRX	EARCRX_DP	DVSS	DVSS	AVDD18_ USB_P0	AVDD18_ SSUSB	AVDD12_ SSUSB		AN
DVSS	DVSS	DVSS	DVSS	DVSS	EARCRX_DM	DVSS	DVSS		DVSS	DVSS	AVDD12_ USB_P0	AP
HDMIRX21_ CH0_M	HDMIRX21_ CH0_P	DVSS	HDMIRX21_ CH2_M	HDMIRX21_ CH2_P	DVSS	HDMITX21_ CH0_M	HDMITX21_ CH0_P	DVSS	HDMITX21_ CH2_M	HDMITX21_ CH2_P	DVSS	AR
		HDMIRX21_ CH1_P			HDMITX21_ CLK_M			HDMITX21_ CH1_M		DVSS	DUMMY	AT
		HDMIRX21_ CH1_M			HDMITX21_ CLK_P			HDMITX21_ CH1_P		DVSS	DUMMY	AU

4.2 Pin Characteristics

Table 4-10 describes the pin characteristics and the multiplexed signals on each ball.

Table 4-10 Pin Characteristics

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
GPIO_00	H31	GPIO0	DIO	0	0		DVDD18	OFF	I
		TP_GPIO0_AO	DIO	1					
		MSDC2_CMD	DIO	2					
		TDMIN_MCK	DIO	3					
		CLKM0	DO	4					
		PERSTN_1	DO	5					
		IDDIG_1P	DI	6					
		DMIC4_CLK	DO	7					
GPIO_01	G34	GPIO1	DIO	0	0		DVDD18	OFF	I
		TP_GPIO1_AO	DIO	1					
		MSDC2_CLK	DIO	2					
		TDMIN_DI	DI	3					
		CLKM1	DO	4					
		CLKREQN_1	DIO	5					
		USB_DRVVBUS_1P	DO	6					
		DMIC4_DAT	DI	7					
GPIO_02	J31	GPIO2	DIO	0	0		DVDD18	OFF	I
		TP_GPIO2_AO	DIO	1					
		MSDC2_DAT3	DIO	2					
		TDMIN_LRCK	DIO	3					
		CLKM2	DO	4					
		WAKEN_1	DI	5					
		DMIC2_CLK	DO	7					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
GPIO_03	G33	GPIO3	DIO	0	0		DVDD18	OFF	I
		TP_GPIO3_AO	DIO	1					
		MSDC2_DAT0	DIO	2					
		TDMIN_BCK	DIO	3					
		CLKM3	DO	4					
		DMIC2_DAT	DI	7					
GPIO_04	G35	GPIO4	DIO	0	0		DVDD18	OFF	I
		TP_GPIO4_AO	DIO	1					
		MSDC2_DAT2	DIO	2					
		SPDIF_IN1	DI	3					
		UTXD3	DO	4					
		SDA2	DIO	5					
		IDDIG_2P	DI	7					
GPIO_05	H33	GPIO5	DIO	0	0		DVDD18	OFF	I
		TP_GPIO5_AO	DIO	1					
		MSDC2_DAT1	DIO	2					
		SPDIF_IN0	DI	3					
		URXD3	DI	4					
		SCL2	DIO	5					
		USB_DRVVBUS_2P	DO	7					
GPIO_06	H32	GPIO6	DIO	0	0		DVDD18	OFF	I
		TP_GPIO6_AO	DIO	1					
		DP_TX_HPD	DI	2					
		I2SO1_D4	DO	3					
		UTXD4	DO	4					
		CMVREF3	DO	5					
		DMIC3_CLK	DO	7					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
GPIO_07	H34	GPIO7	DIO	0	0		DVDD18	OFF	I
		TP_GPIO7_AO	DIO	1					
		EDP_TX_HPD	DI	2					
		I2SO1_D5	DO	3					
		URXD4	DI	4					
		CMVREF4	DO	5					
		DMIC3_DAT	DI	7					
SDA0	D36	GPIO8	DIO	0	1		DVDD18	PU	I
		SDA0	DIO	1					
		PWM_0	DO	2					
		SPDIF_OUT	DO	4					
SCL0	D37	GPIO9	DIO	0	1		DVDD18	PU	I
		SCL0	DIO	1					
		PWM_1	DO	2					
		IR_IN	DI	4					
SDA1	E36	GPIO10	DIO	0	1		DVDD18	PU	I
		SDA1	DIO	1					
		PWM_2	DO	2					
		SPDIF_IN1	DI	4					
SCL1	E35	GPIO11	DIO	0	1		DVDD18	PU	I
		SCL1	DIO	1					
		PWM_3	DO	2					
		SPDIF_IN0	DI	4					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
SDA2	E37	GPIO12	DIO	0	1		DVDD18	PU	I
		SDA2	DIO	1					
		DMIC3_DAT_R	DI	2					
		I2SO1_D6	DO	3					
SCL2	F37	GPIO13	DIO	0	1		DVDD18	PU	I
		SCL2	DIO	1					
		DMIC4_DAT_R	DO	2					
		I2SO1_D7	DI	3					
SDA3	F35	GPIO14	DIO	0	1		DVDD18	PU	I
		SDA3	DIO	1					
		DMIC3_DAT	DI	2					
		TDMIN_MCK	DIO	3					
SCL3	F34	GPIO15	DIO	0	1		DVDD18	PU	I
		SCL3	DIO	1					
		DMIC3_CLK	DI	2					
		TDMIN_DI	DO	3					
SDA4	F36	GPIO16	DIO	0	1		DVDD18	PU	I
		SDA4	DIO	1					
		DMIC4_DAT	DI	2					
		TDMIN_LRCK	DIO	3					
SCL4	G36	GPIO17	DIO	0	1		DVDD18	PU	I
		SCL4	DIO	1					
		DMIC4_CLK	DO	2					
		TDMIN_BCK	DIO	3					
DPTX_HPD	AP25	GPIO18	AIO	0	0		DVDD18_IOBM	OFF	I
		DP_TX_HPD	DI	1					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
PCIE_WAKE_N	AM29	GPIO19	AIO	0	0		DVDD18_IOBM	OFF	I
		WAKEN	DIO	1					
		SCP_SDA1	DIO	2					
		SDA6	DIO	5					
PCIE_PERESET_N	AL28	GPIO20	AIO	0	0		DVDD18_IOBM	OFF	I
		PERSTN	DIO	1					
		SCP_SCL1	DIO	2					
		SCL6	DIO	5					
PCIE_CLKREQ_N	AL29	GPIO21	AIO	0	0		DVDD18_IOBM	OFF	I
		CLKREQN	DIO	1					
		SCP_SDA1	DIO	5					
CMMCLK0	AL27	GPIO22	DIO	0	0		DVDD18	OFF	I
		CMMCLK0	DO	1					
		PERSTN_1	DO	2					
		SCP_SCL1	DIO	5					
CMMCLK1	AK27	GPIO23	DIO	0	0		DVDD18	OFF	I
		CMMCLK1	DO	1					
		CLKREQN_1	DIO	2					
		SDA4	DIO	3					
		DMIC1_CLK	DO	4					
		SCP_SDA0	DIO	5					
CMMCLK2	AM28	GPIO24	DIO	0	0		DVDD18	OFF	I
		CMMCLK2	DO	1					
		WAKEN_1	DI	2					
		SCL4	DIO	3					
		DMIC1_DAT	DI	4					
		SCP_SCL0	DIO	5					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
CMMRST	AL26	GPIO25	DIO	0	0		DVDD18	OFF	I
		CMMRST	DO	1					
		CMMCLK3	DO	2					
		SPDIF_OUT	DO	3					
		SDA6	DIO	4					
CMMPDN	AM26	GPIO26	DIO	0	0		DVDD18	OFF	I
		CMMPDN	DO	1					
		CMMCLK4	DO	2					
		IR_IN	DI	3					
		SCL6	DIO	4					
HDMIRX_HTPLG	AN25	GPIO27	DIO	0	0		DVDD18	OFF	I
		HDMIRX20_HTPLG	DO	1					
		CMFLASH0	DO	2					
		TP_UTXD2_AO	DO	4					
		SCL7	DIO	5					
		UCTS2	DI	6					
HDMIRX_PWR5V	AN24	GPIO28	DIO	0	0		DVDD18	OFF	I
		HDMIRX20_PWR5V	DI	1					
		CMFLASH1	DO	2					
		TP_URXD2_AO	DI	4					
		SDA7	DIO	5					
		URTS2	DO	6					
HDMIRX_SCL	AM25	GPIO29	DIO	0	0		DVDD18	OFF	I
		HDMIRX20_SCL	DI	1					
		CMFLASH2	DO	2					
		SCL5	DIO	3					
		TP_URTS2_AO	DO	4					
		UTXD2	DO	6					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
HDMIRX_SDA	AP24	GPIO30	DIO	0	0		DVDD18	OFF	I
		HDMIRX20_SDA	DIO	1					
		CMFLASH3	DO	2					
		SDA5	DIO	3					
		TP_UCTS2_AO	DI	4					
		URXD2	DI	6					
HDMITX_PWR5V	AL24	GPIO31	DIO	0	0		DVDD18	OFF	I
		HDMITX20_PWR5V	DO	1					
		DMIC1_DAT_R	DI	2					
		PERSTN	DIO	3					
HDMITX_HTPLG	AM24	GPIO32	DIO	0	0		DVDD18	OFF	I
		HDMITX20_HTPLG	DI	1					
		CLKREQN	DIO	3					
HDMITX_CEC	AM23	GPIO33	DIO	0	0		DVDD18	OFF	I
		HDMITX20_CEC	DIO	1					
		CMVREF0	DO	2					
		WAKEN	DIO	3					
HDMITX_SCL	AL23	GPIO34	DIO	0	0		DVDD18	OFF	I
		HDMITX20_SCL	DIO	1					
		CMVREF1	DO	2					
		SCL7	DIO	3					
		SCL6	DIO	4					
HDMITX_SDA	AL25	GPIO35	DIO	0	0		DVDD18	OFF	I
		HDMITX20_SDA	DIO	1					
		CMVREF2	DO	2					
		SDA7	DIO	3					
		SDA6	DIO	4					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
SPIM2_CSB	AN23	GPIO140	DIO	0	1		DVDD18	PU	OH
		SPIM2_CSB	DO	1					
		SPINOR_CS	DO	2					
		SNFI_CS	DO	3					
		DMIC3_DAT	DI	4					
SPIM2_CLK	AP22	GPIO141	DIO	0	0		DVDD18	OFF	I
		SPIM2_CLK	DO	1					
		SPINOR_CK	DO	2					
		SNFI_CLK	DO	3					
		DMIC3_CLK	DO	4					
SPIM2_MO	AN22	GPIO142	DIO	0	0		DVDD18	OFF	I
		SPIM2_MO	DO	1					
		SPINOR_IO0	DIO	2					
		SNFI_MOSI	DIO	3					
		DMIC4_DAT	DI	4					
SPIM2_MI	AP23	GPIO143	DIO	0	1		DVDD18	OFF	I
		SPIM2_MI	DI	1					
		SPINOR_IO1	DIO	2					
		SNFI_MISO	DIO	3					
		DMIC4_CLK	DO	4					
SPIM1_CSB	AT22	GPIO136	DIO	0	0		DVDD18	OFF	I
		SPIM1_CSB	DO	1					
		SCP_SPI1_A_CS	DO	2					
		SPIS1_CSB	DI	3					
SPIM1_CLK	AU23	GPIO137	DIO	0	0		DVDD18	OFF	I
		SPIM1_CLK	DO	1					
		SCP_SPI1_A_CK	DO	2					
		SPIS1_CLK	DI	3					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
SPIM1_MO	AT23	GPIO138	DIO	0	0		DVDD18	OFF	I
		SPIM1_MO	DO	1					
		SCP_SPI1_A_MO	DO	2					
		SPIS1_SI	DI	3					
SPIM1_MI	AR23	GPIO139	DIO	0	0		DVDD18	OFF	I
		SPIM1_MI	DI	1					
		SCP_SPI1_A_MI	DI	2					
		SPIS1_SO	DO	3					
PMIC_RTC32K_CK	AR21	GPIO36	DIO	0	1		DVDD18	OFF	I
		RTC32K_CK	DI	1					
PMIC_WATCHDOG	AM22	GPIO37	DIO	0	1		DVDD18	OFF	OL
		PMIC_WATCHDOG	DO	1					
PMIC_SRCLKEN_IN0	AM21	GPIO38	DIO	0	1		DVDD18	PU	OH
		SRCLKENA0	DO	1					
PMIC_SRCLKEN_IN1	AL21	GPIO39	DIO	0	1		DVDD18	PU	OH
		SRCLKENA1	DO	1					
		DMIC2_DAT_R	DI	2					
PWRAP_SPI_CSN	AU21	GPIO40	DIO	0	1		DVDD18	PU	OH
		PWRAP_SPI0_CSN	DO	1					
		SPIM3_CSB	DO	3					
PWRAP_SPI_CK	AT21	GPIO41	DIO	0	1		DVDD18	OFF	OL
		PWRAP_SPI0_CK	DO	1					
		SPIM3_CLK	DO	3					
PWRAP_SPI_MO	AP21	GPIO42	DIO	0	1		DVDD18	OFF	I
		PWRAP_SPI0_MO	DIO	1					
		PWRAP_SPI0_MI	DIO	2					
		SPIM3_MO	DO	3					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
PWRAP_SPI_MI	AN21	GPIO43	DIO	0	1		DVDD18	OFF	I
		PWRAP_SPI0_MI	DIO	1					
		PWRAP_SPI0_MO	DIO	2					
		SPIM3_MI	DI	3					
SPMI_M_SCL	AN20	GPIO44	DIO	0	1		DVDD18	OFF	OL
		SPMI_M_SCL	DIO	1					
		SCL5	DIO	3					
		UTXD5	DO	4					
SPMI_M_SDA	AM20	GPIO45	DIO	0	1		DVDD18	OFF	I
		SPMI_M_SDA	DIO	1					
		SDA5	DIO	3					
		URXD5	DI	4					
I2SIN_MCK	AL20	GPIO46	DIO	0	0		DVDD18	OFF	I
		I2SIN_MCK	DIO	1					
		SPLIN_MCK	DI	3					
I2SIN_BCK	AK20	GPIO47	DIO	0	0		DVDD18	OFF	I
		I2SIN_BCK	DIO	1					
		SPLIN_LRCK	DI	3					
I2SIN_WS	AR20	GPIO48	DIO	0	0		DVDD18	OFF	I
		I2SIN_WS	DIO	1					
		SPLIN_BCK	DI	3					
I2SIN_D0	AP20	GPIO49	DIO	0	0		DVDD18	OFF	I
		I2SIN_D0	DI	1					
		SPLIN_D0	DI	3					
I2SO1_MCK	AN19	GPIO50	DIO	0	0		DVDD18	OFF	I
		I2SO1_MCK	DO	1					
I2SO1_BCK	AK19	GPIO51	DIO	0	0		DVDD18	OFF	I
		I2SO1_BCK	DO	1					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
I2SO1_WS	AT19	GPIO52	DIO	0	0		DVDD18	OFF	I
		I2SO1_WS	DO	1					
I2SO1_D0	AR18	GPIO53	DIO	0	0		DVDD18	OFF	I
		I2SO1_D0	DO	1					
I2SO1_D1	AP18	GPIO54	DIO	0	0		DVDD18	OFF	I
		I2SO1_D1	DO	1					
		SPLIN_D1	DI	3					
I2SO1_D2	AN18	GPIO55	DIO	0	0		DVDD18	OFF	I
		I2SO1_D2	DO	1					
		SPLIN_D2	DI	3					
I2SO1_D3	AM18	GPIO56	DIO	0	0		DVDD18	OFF	I
		I2SO1_D3	DO	1					
		SPLIN_D3	DI	3					
I2SO2_MCK	AL18	GPIO57	DIO	0	0		DVDD18	OFF	I
		I2SO2_MCK	DO	1					
		LCM1_RST	DO	3					
I2SO2_BCK	AK18	GPIO58	DIO	0	0		DVDD18	OFF	I
		I2SO2_BCK	DIO	1					
I2SO2_WS	AU20	GPIO59	DIO	0	0		DVDD18	OFF	I
		I2SO2_WS	DIO	1					
I2SO2_D0	AT20	GPIO60	DIO	0	0		DVDD18	OFF	I
		I2SO2_D0	DO	1					
DMIC1_SCK	AP17	GPIO61	DIO	0	0		DVDD18_IOBM	OFF	I
		DMIC1_CLK	DO	1					
		I2SO2_BCK	DIO	2					
		SCP_SPI2_CK	DO	3					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
DMIC1_DAT	AL17	GPIO62	DIO	0	0		DVDD18_IOBM	OFF	I
		DMIC1_DAT	DI	1					
		I2SO2_WS	DIO	2					
		SCP_SPI2_MI	DI	3					
DMIC2_SCK	AU18	GPIO63	DIO	0	0		DVDD18_IOBM	OFF	I
		DMIC2_CLK	DO	1					
		VBUSVALID	DI	2					
		SCP_SPI2_MO	DO	3					
		SCP_SCL2	DIO	4					
DMIC2_DAT	AT18	GPIO64	DIO	0	0		DVDD18_IOBM	OFF	I
		DMIC2_DAT	DI	1					
		VBUSVALID_1P	DI	2					
		SCP_SPI2_CS	DO	3					
		SCP_SDA2	DIO	4					
PCM_DO	AR16	GPIO65	DO	0	0		DVDD18_IOBM	OFF	I
		PCM_DO	DO	1					
		AUXIF_ST0	DO	2					
		UCTS2	DI	3					
PCM_CLK	AP16	GPIO66	DIO	0	0		DVDD18_IOBM	OFF	I
		PCM_CLK	DIO	1					
		AUXIF_CLK0	DO	2					
		URTS2	DO	3					
PCM_DI	AN16	GPIO67	DI	0	0		DVDD18_IOBM	OFF	I
		PCM_DI	DI	1					
		AUXIF_ST1	DO	2					
		UTXD2	DO	3					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
PCM_SYNC	AM16	GPIO68	DIO	0	0		DVDD18_IOBM	OFF	I
		PCM_SYNC	DIO	1					
		AUXIF_CLK1	DO	2					
		URXD2	DI	3					
AUD_CLK_MOSI	AL16	GPIO69	DIO	0	0		DVDD18	OFF	I
		AUD_CLK_MOSI	DO	1					
		PWM_0	DO	3					
		WAKEN	DIO	4					
AUD_SYNC_MOSI	AK16	GPIO70	DIO	0	0		DVDD18	OFF	I
		AUD_SYNC_MOSI	DO	1					
		PWM_1	DO	3					
		PERSTN	DIO	4					
AUD_DAT_MOSI0	AT16	GPIO71	DIO	0	0		DVDD18	OFF	I
		AUD_DAT_MOSI0	DO	1					
		IDDIG_2P	DI	2					
		PWM_2	DO	3					
		CLKREQN	DIO	4					
AUD_DAT_MOSI1	AN15	GPIO72	DIO	0	0		DVDD18	OFF	I
		AUD_DAT_MOSI1	DO	1					
		USB_DRVVBUS_2P	DO	2					
		PWM_3	DO	3					
		PERSTN_1	DO	4					
AUD_DAT_MISO0	AK15	GPIO73	DIO	0	0		DVDD18	OFF	I
		AUD_DAT_MISO0	DI	1					
		CLKREQN_1	DIO	4					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
AUD_DAT_MISO1	AU17	GPIO74	DIO	0	0		DVDD18	OFF	I
		AUD_DAT_MISO1	DI	1					
		WAKEN_1	DI	4					
AUD_DAT_MISO2	AT17	GPIO75	DIO	0	0		DVDD18	OFF	I
		AUD_DAT_MISO2	DI	1					
SCP_VREQ_VAO	AT15	GPIO76	DIO	0	0		DVDD18	OFF	I
		SCP_VREQ_VAO	DO	1					
DGI_D0	AN14	GPIO77	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D0	DO	2					
		SPIM4_CLK	DO	4					
		GBE_TXD3	DO	5					
DGI_D1	AM14	GPIO78	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D1	DO	2					
		SPIM4_MO	DO	4					
		GBE_TXD2	DO	5					
DGI_D2	AL14	GPIO79	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D2	DO	2					
		SPIM4_CSB	DO	4					
		GBE_TXD1	DO	5					
DGI_D3	AK14	GPIO80	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D3	DO	2					
		SPIM4_MI	DI	4					
		GBE_TXD0	DO	5					
DGI_D4	AU15	GPIO81	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D4	DO	2					
		SPIM5_CLK	DO	4					
		GBE_RXD3	DI	5					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
DGI_D5	AR14	GPIO82	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D5	DO	2					
		SPIM5_MO	DO	4					
		GBE_RXD2	DI	5					
DGI_D6	AK11	GPIO83	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D6	DO	2					
		SPIM5_CSB	DO	4					
		GBE_RXD1	DI	5					
DGI_D7	AK10	GPIO84	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D7	DO	2					
		SPIM5_MI	DI	4					
		GBE_RXD0	DI	5					
DGI_D8	AP14	GPIO85	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D8	DO	2					
		SCP_SPI1_B_CK	DO	4					
		GBE_TXC	DIO	5					
DGI_D9	AR12	GPIO86	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D9	DO	2					
		SCP_SPI1_B_MI	DI	4					
		GBE_RXC	DI	5					
DGI_D10	AL12	GPIO87	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D10	DO	2					
		SCP_SPI1_B_CS	DO	4					
		GBE_RXDV	DI	5					
DGI_D11	AK12	GPIO88	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D11	DO	2					
		SCP_SPI1_B_MO	DO	4					
		GBE_TXEN	DO	5					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
DGI_D12	AP12	GPIO89	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D12	DO	2					
		MSDC2_CMD_A	DIO	3					
		GBE_MDC	DO	5					
DGI_D13	AL13	GPIO90	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D13	DO	2					
		MSDC2_CLK_A	DIO	3					
		GBE_MDIO	DIO	5					
DGI_D14	AP13	GPIO91	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D14	DO	2					
		MSDC2_DAT3_A	DIO	3					
		GBE_TXER	DO	5					
DGI_D15	AN12	GPIO92	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_D15	DO	2					
		MSDC2_DAT0_A	DIO	3					
		I2SO2_D1	DO	4					
		GBE_RXER	DI	5					
DGI_HSYNC	AU14	GPIO93	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_HSYNC	DO	2					
		MSDC2_DAT2_A	DIO	3					
		I2SO2_D2	DO	4					
		GBE_COL	DI	5					
DGI_VSYNC	AT13	GPIO94	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_VSYNC	DO	2					
		MSDC2_DAT1_A	DIO	3					
		I2SO2_D3	DO	4					
		GBE_INTR	DI	5					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
DGI_DE	AT12	GPIO95	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_DE	DO	2					
		UTXD2	DO	3					
		I2SIN_D1	DI	5					
DGI_CK	AT14	GPIO96	DIO	0	0		DVDD28_IODGI	OFF	I
		DPI_CK	DO	2					
		URXD2	DI	3					
		I2SIN_D2	DI	5					
DISP_PWM0	AP9	GPIO97	DIO	0	0		DVDD18	OFF	I
		DISP_PWM0	DO	1					
UART0_TXD	AR10	GPIO98	DIO	0	1		DVDD18	PU	OH
		UTXD0	DO	1					
UART0_RXD	AP10	GPIO99	DIO	0	1		DVDD18	PU	I
		URXD0	DI	1					
UART1_RTS	AM8	GPIO100	DIO	0	0		DVDD18	OFF	I
		URTS1	DO	1					
		DSI_TE	DI	2					
		I2SO1_D8	DO	3					
		KPROW2	DIO	4					
		PWM_0	DO	5					
		TP_URTS1_AO	DO	6					
		I2SIN_D0	DI	7					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
UART1_CTS	AT10	GPIO101	DIO	0	0		DVDD18	OFF	I
		UCTS1	DI	1					
		DSI1_TE	DI	2					
		I2SO1_D9	DO	3					
		KPCOL2	DIO	4					
		PWM_1	DO	5					
		TP_UCTS1_AO	DI	6					
		I2SIN_D1	DI	7					
UART1_TXD	AR8	GPIO102	DIO	0	0		DVDD18	OFF	I
		UTXD1	DO	1					
		VBUSVALID_2P	DI	2					
		I2SO1_D10	DO	3					
		TP_UTXD1_AO	DO	5					
		I2SIN_D2	DI	7					
UART1_RXD	AP8	GPIO103	DIO	0	0		DVDD18	OFF	I
		URXD1	DI	1					
		VBUSVALID_3P	DI	2					
		I2SO1_D11	DO	3					
		TP_URXD1_AO	DI	5					
		I2SIN_D3	DI	7					
KPROW0	AN11	GPIO104	DIO	0	1		DVDD18	OFF	OL
		KPROW0	DIO	1					
		DISP_PWM1	DO	2					
KPROW1	AN10	GPIO105	DIO	0	0		DVDD18	OFF	I
		KPROW1	DIO	1					
		EDP_TX_HPD	DI	2					
		PWM_2	DO	3					
KPCOL0	AU11	GPIO106	DIO	0	1		DVDD18	PU	I
		KPCOL0	DIO	1					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
KPCOL1	AT11	GPIO107	DIO	0	0		DVDD18	OFF	I
		KPCOL1	DIO	1					
		DSI1_TE	DI	2					
		PWM_3	DO	3					
		SCP_SCL3	DIO	4					
		I2SIN_MCK	DIO	5					
DSI_LCM_RST	AL10	GPIO108	DIO	0	0		DVDD18	OFF	I
		LCM_RST	DO	1					
		KPCOL1	DIO	2					
		SCP_SDA3	DIO	4					
		I2SIN_BCK	DIO	5					
DSI_DSI_TE	AT9	GPIO109	DIO	0	0		DVDD18	OFF	I
		DSI_TE	DI	1					
		I2SIN_D3	DI	2					
		I2SIN_WS	DIO	5					
USB_IDDIG	AL9	GPIO128	AIO	0	0		DVDD18_IOBR	OFF	I
		IDDIG	DI	1					
		UCTS2	DI	2					
		UTXD5	DO	3					
		UFS_MPHY_SCL	DI	4					
		SCP_SCL2	DIO	7					
USB_DRV_VBUS	AM10	GPIO129	AIO	0	0		DVDD18_IOBR	OFF	I
		USB_DRVVBUS	DO	1					
		URTS2	DO	2					
		URXD5	DI	3					
		UFS_MPHY_SDA	DIO	4					
		SCP_SDA2	DIO	7					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
USB_IDDIG_1P	AL7	GPIO130	AIO	0	0		DVDD18_IOBR	OFF	I
		IDDIG_1P	DI	1					
		SPINOR_IO2	DIO	2					
		SNFI_WP	DIO	3					
USB_DRV_VBUS_1P	AL8	GPIO131	AIO	0	0		DVDD18_IOBR	OFF	I
		USB_DRVVBUS_1P	DO	1					
		SPINOR_IO3	DIO	2					
		SNFI_HOLD	DIO	3					
MSDC1_CMD	AN8	GPIO110	DIO	0	1	MSDC1IO	DVDD28_MSDC1	PU	I
		MSDC1_CMD	DIO	1					
MSDC1_CLK	AU9	GPIO111	DIO	0	1	MSDC1IO	DVDD28_MSDC1	OFF	OL
		MSDC1_CLK	DIO	1					
MSDC1_DAT0	AT6	GPIO112	DIO	0	1	MSDC1IO	DVDD28_MSDC1	PU	I
		MSDC1_DAT0	DIO	1					
		I2SO2_D0	DO	4					
MSDC1_DAT1	AT7	GPIO113	DIO	0	1	MSDC1IO	DVDD28_MSDC1	PU	I
		MSDC1_DAT1	DIO	1					
		I2SO2_D1	DO	4					
MSDC1_DAT2	AT8	GPIO114	DIO	0	1	MSDC1IO	DVDD28_MSDC1	PU	I
		MSDC1_DAT2	DIO	1					
		I2SO2_D2	DO	4					
MSDC1_DAT3	AU8	GPIO115	DIO	0	1	MSDC1IO	DVDD28_MSDC1	PU	I
		MSDC1_DAT3	DIO	1					
		I2SO2_D3	DO	4					
EMMC_DAT7	E32	GPIO116	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT7	DIO	1					
EMMC_DAT6	E33	GPIO117	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT6	DIO	1					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
EMMC_DAT5	E34	GPIO118	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT5	DIO	1					
EMMC_DAT4	D35	GPIO119	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT4	DIO	1					
EMMC_RSTB	B35	GPIO120	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	OH
		MSDC0_RSTB	DO	1					
EMMC_CMD	C36	GPIO121	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_CMD	DIO	1					
EMMC_CLK	B36	GPIO122	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	OFF	OL
		MSDC0_CLK	DIO	1					
EMMC_DAT3	C35	GPIO123	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT3	DIO	1					
EMMC_DAT2	D32	GPIO124	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT2	DIO	1					
EMMC_DAT1	D33	GPIO125	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT1	DIO	1					
EMMC_DAT0	D34	GPIO126	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	PU	I
		MSDC0_DAT0	DIO	1					
EMMC_DSL	A35	GPIO127	DIO	0	1	MSDC0IO	DVDD18_IOEMMC	OFF	I
		MSDC0_DSL	DIO	1					
SPIM0_CSB	F31	GPIO132	DIO	0	0		DVDD18	OFF	I
		SPIM0_CSB	DO	1					
		SCP_SPI0_CS	DO	2					
		SPIS0_CSB	DI	3					
SPIM0_CLK	F32	GPIO133	DIO	0	0		DVDD18	OFF	I
		SPIM0_CLK	DO	1					
		SCP_SPI0_CK	DO	2					
		SPIS0_CLK	DI	3					

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
SPIM0_MO	G32	GPIO134	DIO	0	0		DVDD18	OFF	I
		SPIM0_MO	DO	1					
		SCP_SPI0_MO	DO	2					
		SPIS0_SI	DI	3					
SPIM0_MI	G31	GPIO135	DIO	0	0		DVDD18	OFF	I
		SPIM0_MI	DI	1					
		SCP_SPI0_MI	DI	2					
		SPIS0_SO	DO	3					
SYSRSTB	AL22	SYSRSTB	DI				DVDD18		
TESTMODE	AK22	TESTMODE	DI				DVDD18		
X26M_IN	AB28	X26M_IN	AI				AVDD12_CKSQ		
EMI0_CA2	A18	EMI0_CA2	DIO			DDRIO	AVDDQ_EMI		
EMI0_CA4	A20	EMI0_CA4	DIO			DDRIO	AVDDQ_EMI		
EMI0_DMI1	A22	EMI0_DMI1	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ8	A24	EMI0_DQ8	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ7	A26	EMI0_DQ7	DIO			DDRIO	AVDDQ_EMI		
EMI0_DMI0	A28	EMI0_DMI0	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ2	A30	EMI0_DQ2	DIO			DDRIO	AVDDQ_EMI		
EMI0_EXTR	B3	EMI0_EXTR	DIO			DDRIO	AVDDQ_EMI		
EMI0_CA1	B17	EMI0_CA1	DIO			DDRIO	AVDDQ_EMI		
EMI0_CA0	B18	EMI0_CA0	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ14	B21	EMI0_DQ14	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ9	B23	EMI0_DQ9	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ15	B25	EMI0_DQ15	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ6	B27	EMI0_DQ6	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ5	B29	EMI0_DQ5	DIO			DDRIO	AVDDQ_EMI		
EMI0_CA3	C20	EMI0_CA3	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ4	C26	EMI0_DQ4	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQS0_T	C28	EMI0_DQS0_T	DIO			DDRIO	AVDDQ_EMI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
EMI0_CA5	D19	EMI0_CA5	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ12	D21	EMI0_DQ12	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ13	D22	EMI0_DQ13	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQS0_C	D28	EMI0_DQS0_C	DIO			DDRIO	AVDDQ_EMI		
EMI0_RESET_N	D29	EMI0_RESET_N	DIO			DDRIO	AVDD12_EMI		
EMI0_CK_T	E17	EMI0_CK_T	DIO			DDRIO	AVDDQ_EMI		
EMI0_CS1	E18	EMI0_CS1	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ11	E23	EMI0_DQ11	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQS1_T	E25	EMI0_DQS1_T	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ3	E26	EMI0_DQ3	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ0	E27	EMI0_DQ0	DIO			DDRIO	AVDDQ_EMI		
EMI0_CK_C	F17	EMI0_CK_C	DIO			DDRIO	AVDDQ_EMI		
EMI0_CS0	F19	EMI0_CS0	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ10	F24	EMI0_DQ10	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQS1_C	F25	EMI0_DQS1_C	DIO			DDRIO	AVDDQ_EMI		
EMI0_DQ1	F26	EMI0_DQ1	DIO			DDRIO	AVDDQ_EMI		
EMI0_CKE1	G20	EMI0_CKE1	DIO			DDRIO	AVDD12_EMI		
EMI0_CKE0	G21	EMI0_CKE0	DIO			DDRIO	AVDD12_EMI		
EMI0_TP	H19	EMI0_TP	DIO			DDRIO	AVDD12_EMI		
EMI1_DQ2	A3	EMI1_DQ2	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ5	A4	EMI1_DQ5	DIO			DDRIO	AVDDQ_EMI		
EMI1_DMI0	A5	EMI1_DMI0	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ7	A7	EMI1_DQ7	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ8	A9	EMI1_DQ8	DIO			DDRIO	AVDDQ_EMI		
EMI1_DMI1	A11	EMI1_DMI1	DIO			DDRIO	AVDDQ_EMI		
EMI1_CA4	A13	EMI1_CA4	DIO			DDRIO	AVDDQ_EMI		
EMI1_CA2	A15	EMI1_CA2	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ6	B6	EMI1_DQ6	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ15	B8	EMI1_DQ15	DIO			DDRIO	AVDDQ_EMI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
EMI1_DQ9	B10	EMI1_DQ9	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ14	B12	EMI1_DQ14	DIO			DDRIO	AVDDQ_EMI		
EMI1_CA5	B15	EMI1_CA5	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQS0_T	C5	EMI1_DQS0_T	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ0	C6	EMI1_DQ0	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ13	C12	EMI1_DQ13	DIO			DDRIO	AVDDQ_EMI		
EMI1_CA3	C13	EMI1_CA3	DIO			DDRIO	AVDDQ_EMI		
EMI1_CKE1	C14	EMI1_CKE1	DIO			DDRIO	AVDD12_EMI		
EMI1_CA1	C16	EMI1_CA1	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQS0_C	D5	EMI1_DQS0_C	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ3	D8	EMI1_DQ3	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQS1_T	D9	EMI1_DQS1_T	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ10	D10	EMI1_DQ10	DIO			DDRIO	AVDDQ_EMI		
EMI1_CKE0	D14	EMI1_CKE0	DIO			DDRIO	AVDD12_EMI		
EMI1_CK_C	D16	EMI1_CK_C	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ1	E7	EMI1_DQ1	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ4	E8	EMI1_DQ4	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQS1_C	E9	EMI1_DQS1_C	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ12	E12	EMI1_DQ12	DIO			DDRIO	AVDDQ_EMI		
EMI1_CA0	E15	EMI1_CA0	DIO			DDRIO	AVDDQ_EMI		
EMI1_CK_T	E16	EMI1_CK_T	DIO			DDRIO	AVDDQ_EMI		
EMI1_DQ11	F10	EMI1_DQ11	DIO			DDRIO	AVDDQ_EMI		
EMI1_CS0	F14	EMI1_CS0	DIO			DDRIO	AVDDQ_EMI		
EMI1_CS1	F15	EMI1_CS1	DIO			DDRIO	AVDDQ_EMI		
EMI2_RESET_N	B2	EMI2_RESET_N	DIO			DDRIO	AVDD12_EMI		
EMI2_DQ0	C1	EMI2_DQ0	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQS0_T	C3	EMI2_DQS0_T	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ1	D2	EMI2_DQ1	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQS0_C	D3	EMI2_DQS0_C	DIO			DDRIO	AVDDQ_EMI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
EMI2_DMI0	E1	EMI2_DMI0	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ5	E4	EMI2_DQ5	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ6	F2	EMI2_DQ6	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ2	F3	EMI2_DQ2	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ4	F5	EMI2_DQ4	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ7	G1	EMI2_DQ7	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ3	G5	EMI2_DQ3	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ15	H2	EMI2_DQ15	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQS1_T	H4	EMI2_DQS1_T	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQS1_C	H5	EMI2_DQS1_C	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ8	J1	EMI2_DQ8	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ9	K2	EMI2_DQ9	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ11	K4	EMI2_DQ11	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ10	K5	EMI2_DQ10	DIO			DDRIO	AVDDQ_EMI		
EMI2_DMI1	L1	EMI2_DMI1	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ13	L4	EMI2_DQ13	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ14	M2	EMI2_DQ14	DIO			DDRIO	AVDDQ_EMI		
EMI2_DQ12	M3	EMI2_DQ12	DIO			DDRIO	AVDDQ_EMI		
EMI2_CS0	M5	EMI2_CS0	DIO			DDRIO	AVDDQ_EMI		
EMI2_CA4	N1	EMI2_CA4	DIO			DDRIO	AVDDQ_EMI		
EMI2_CA3	N3	EMI2_CA3	DIO			DDRIO	AVDDQ_EMI		
EMI2_CS1	N4	EMI2_CS1	DIO			DDRIO	AVDDQ_EMI		
EMI2_CA0	N5	EMI2_CA0	DIO			DDRIO	AVDDQ_EMI		
EMI2_CA5	P3	EMI2_CA5	DIO			DDRIO	AVDDQ_EMI		
EMI2_CA2	P4	EMI2_CA2	DIO			DDRIO	AVDDQ_EMI		
EMI2_CA1	P5	EMI2_CA1	DIO			DDRIO	AVDDQ_EMI		
EMI2_CKE1	R1	EMI2_CKE1	DIO			DDRIO	AVDD12_EMI		
EMI2_CKE0	R2	EMI2_CKE0	DIO			DDRIO	AVDD12_EMI		
EMI2_CK_T	R4	EMI2_CK_T	DIO			DDRIO	AVDDQ_EMI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
EMI2_CK_C	R5	EMI2_CK_C	DIO			DDRIO	AVDDQ_EMI		
EMI2_TP	R7	EMI2_TP	DIO			DDRIO	AVDD12_EMI		
EMI2_EXTR	AJ1	EMI2_EXTR	DIO			DDRIO	AVDDQ_EMI		
EMI3_CK_T	T4	EMI3_CK_T	DIO			DDRIO	AVDDQ_EMI		
EMI3_CK_C	T5	EMI3_CK_C	DIO			DDRIO	AVDDQ_EMI		
EMI3_CKE1	U1	EMI3_CKE1	DIO			DDRIO	AVDD12_EMI		
EMI3_CKE0	U2	EMI3_CKE0	DIO			DDRIO	AVDD12_EMI		
EMI3_CA2	U3	EMI3_CA2	DIO			DDRIO	AVDDQ_EMI		
EMI3_CA1	U5	EMI3_CA1	DIO			DDRIO	AVDDQ_EMI		
EMI3_CA0	U6	EMI3_CA0	DIO			DDRIO	AVDDQ_EMI		
EMI3_CA5	V3	EMI3_CA5	DIO			DDRIO	AVDDQ_EMI		
EMI3_CS1	V4	EMI3_CS1	DIO			DDRIO	AVDDQ_EMI		
EMI3_CS0	V5	EMI3_CS0	DIO			DDRIO	AVDDQ_EMI		
EMI3_CA4	W1	EMI3_CA4	DIO			DDRIO	AVDDQ_EMI		
EMI3_CA3	W3	EMI3_CA3	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ14	Y2	EMI3_DQ14	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ13	Y3	EMI3_DQ13	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ12	Y5	EMI3_DQ12	DIO			DDRIO	AVDDQ_EMI		
EMI3_DMI1	AA1	EMI3_DMI1	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ11	AA5	EMI3_DQ11	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ9	AB2	EMI3_DQ9	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ10	AB4	EMI3_DQ10	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ8	AC1	EMI3_DQ8	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQS1_T	AC4	EMI3_DQS1_T	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQS1_C	AC5	EMI3_DQS1_C	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ15	AD2	EMI3_DQ15	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ3	AD4	EMI3_DQ3	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ7	AE1	EMI3_DQ7	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ4	AE5	EMI3_DQ4	DIO			DDRIO	AVDDQ_EMI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
EMI3_DQ6	AF2	EMI3_DQ6	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ5	AF3	EMI3_DQ5	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ2	AF5	EMI3_DQ2	DIO			DDRIO	AVDDQ_EMI		
EMI3_DMI0	AG1	EMI3_DMI0	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQS0_C	AG3	EMI3_DQS0_C	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQS0_T	AG4	EMI3_DQS0_T	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ1	AH1	EMI3_DQ1	DIO			DDRIO	AVDDQ_EMI		
EMI3_DQ0	AH2	EMI3_DQ0	DIO			DDRIO	AVDDQ_EMI		
DSI0_D2P_T0A	AJ3	DSI0_D2P_T0A	AIO				AVDD12_DSI		
DSI0_D2N_T0B	AJ4	DSI0_D2N_T0B	AIO				AVDD12_DSI		
DSI0_D0N_T1A	AK3	DSI0_D0N_T1A	AIO				AVDD12_DSI		
DSI0_D0P_T0C	AK4	DSI0_D0P_T0C	AIO				AVDD12_DSI		
DSI0_CKN_T1C	AL2	DSI0_CKN_T1C	AIO				AVDD12_DSI		
DSI0_D1P_T2A	AL3	DSI0_D1P_T2A	AIO				AVDD12_DSI		
DSI0_CKP_T1B	AM2	DSI0_CKP_T1B	AIO				AVDD12_DSI		
DSI0_D1N_T2B	AM3	DSI0_D1N_T2B	AIO				AVDD12_DSI		
DSI0_D3P_T2C	AM4	DSI0_D3P_T2C	AIO				AVDD12_DSI		
DSI0_D3N	AN4	DSI0_D3N	AIO				AVDD12_DSI		
DSI1_D2N_T0B	AN1	DSI1_D2N_T0B	AO				AVDD12_CSI		
DSI1_D2P_T0A	AN2	DSI1_D2P_T0A	AIO				AVDD12_CSI		
DSI1_D0P_T0C	AP2	DSI1_D0P_T0C	AIO				AVDD12_CSI		
DSI1_D0N_T1A	AR1	DSI1_D0N_T1A	AIO				AVDD12_CSI		
DSI1_CKP_T1B	AR2	DSI1_CKP_T1B	AIO				AVDD12_CSI		
DSI1_CKN_T1C	AR3	DSI1_CKN_T1C	AIO				AVDD12_CSI		
DSI1_D3N	AR4	DSI1_D3N	AIO				AVDD12_CSI		
DSI1_D3P_T2C	AT4	DSI1_D3P_T2C	AIO				AVDD12_CSI		
DSI1_D1P_T2A	AU3	DSI1_D1P_T2A	AIO				AVDD12_CSI		
DSI1_D1N_T2B	AU4	DSI1_D1N_T2B	AIO				AVDD12_CSI		
CSI0A_L0P	J35	CSI0A_L0P	AIO				AVDD12_CSI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
CSI0A_L0N	J36	CSI0A_L0N	AIO				AVDD12_CSI		
CSI0A_L2N	K33	CSI0A_L2N	AIO				AVDD12_CSI		
CSI0A_L2P	K34	CSI0A_L2P	AIO				AVDD12_CSI		
CSI0A_L1P	K35	CSI0A_L1P	AIO				AVDD12_CSI		
CSI0B_L0P	L32	CSI0B_L0P	AIO				AVDD12_CSI		
CSI0B_L0N	L33	CSI0B_L0N	AIO				AVDD12_CSI		
CSI0A_L1N	L34	CSI0A_L1N	AIO				AVDD12_CSI		
CSI0D_L0P	L36	CSI0D_L0P	AIO				AVDD12_CSI		
CSI0D_L0N	L37	CSI0D_L0N	AIO				AVDD12_CSI		
CSI0B_L1N	M32	CSI0B_L1N	AIO				AVDD12_CSI		
CSI0B_L1P	M33	CSI0B_L1P	AIO				AVDD12_CSI		
CSI0C_L0N	M35	CSI0C_L0N	AIO				AVDD12_CSI		
CSI0D_L1N	M36	CSI0D_L1N	AIO				AVDD12_CSI		
CSI0D_L1P	M37	CSI0D_L1P	AIO				AVDD12_CSI		
CSI0C_L2P	N31	CSI0C_L2P	AIO				AVDD12_CSI		
CSI0C_L2N	N32	CSI0C_L2N	AIO				AVDD12_CSI		
CSI0C_L1N	N33	CSI0C_L1N	AIO				AVDD12_CSI		
CSI0C_L1P	N34	CSI0C_L1P	AIO				AVDD12_CSI		
CSI0C_L0P	N35	CSI0C_L0P	AIO				AVDD12_CSI		
CSI1A_L0P_T0A	R32	CSI1A_L0P_T0A	AIO				AVDD12_CSI		
CSI1A_L0N_T0B	R33	CSI1A_L0N_T0B	AIO				AVDD12_CSI		
CSI1A_L1P_T0C	R34	CSI1A_L1P_T0C	AIO				AVDD12_CSI		
CSI1A_L1N_T1A	R35	CSI1A_L1N_T1A	AIO				AVDD12_CSI		
CSI1A_L2P_T1B	R36	CSI1A_L2P_T1B	AIO				AVDD12_CSI		
CSI1A_L2N_T1C	R37	CSI1A_L2N_T1C	AIO				AVDD12_CSI		
CSI1B_L2N_T1C	T32	CSI1B_L2N_T1C	AIO				AVDD12_CSI		
CSI1B_L2P_T1B	T33	CSI1B_L2P_T1B	AIO				AVDD12_CSI		
CSI1B_L1N_T1A	T34	CSI1B_L1N_T1A	AIO				AVDD12_CSI		
CSI1B_L1P_T0C	T35	CSI1B_L1P_T0C	AIO				AVDD12_CSI		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
CSI1B_L0P_T0A	T36	CSI1B_L0P_T0A	AIO				AVDD12_CSI		
CSI1B_L0N_T0B	T37	CSI1B_L0N_T0B	AIO				AVDD12_CSI		
SSUSB_TXN	AH32	SSUSB_TXN	AIO				AVDD12_SSUSB		
SSUSB_TXP	AH33	SSUSB_TXP	AIO				AVDD12_SSUSB		
SSUSB_RXP	AJ31	SSUSB_RXP	AIO				AVDD12_SSUSB		
SSUSB_RXN	AK31	SSUSB_RXN	AIO				AVDD12_SSUSB		
PCIE_LN0_RXP_P1	V33	PCIE_LN0_RXP_P1	AIO				AVDD12_PCIE_P1		
PCIE_LN0_RXN_P1	V34	PCIE_LN0_RXN_P1	AIO				AVDD12_PCIE_P1		
PCIE_CKRXP_P1	V36	PCIE_CKRXP_P1	AIO				AVDD12_PCIE_P1		
PCIE_CKRXN_P1	V37	PCIE_CKRXN_P1	AIO				AVDD12_PCIE_P1		
PCIE_LN0_TXP_P1	W31	PCIE_LN0_TXP_P1	AIO				AVDD12_PCIE_P1		
PCIE_LN0_TXN_P1	W32	PCIE_LN0_TXN_P1	AIO				AVDD12_PCIE_P1		
PCIEG3_LN0_RXP	Y35	PCIEG3_LN0_RXP	AIO				AVDD12_PCIEG3		
PCIEG3_LN0_RXN	W35	PCIEG3_LN0_RXN	AIO				AVDD12_PCIEG3		
PCIEG3_LN0_TXP	AA30	PCIEG3_LN0_TXP	AIO				AVDD12_PCIEG3		
PCIEG3_LN0_TXN	AA31	PCIEG3_LN0_TXN	AIO				AVDD12_PCIEG3		
PCIEG3_LN1_RXP	AC35	PCIEG3_LN1_RXP	AIO				AVDD12_PCIEG3		
PCIEG3_LN1_RXN	AB35	PCIEG3_LN1_RXN	AIO				AVDD12_PCIEG3		
PCIEG3_LN1_TXP	AA37	PCIEG3_LN1_TXP	AIO				AVDD12_PCIEG3		
PCIEG3_LN1_TXN	AA36	PCIEG3_LN1_TXN	AIO				AVDD12_PCIEG3		
PCIEG3_CLKP	AA34	PCIEG3_CLKP	AIO				AVDD12_PCIEG3		
PCIEG3_CLKN	AA33	PCIEG3_CLKN	AIO				AVDD12_PCIEG3		
DP_LN0_TXP	AG35	DP_LN0_TXP	AIO				AVDD12_DPTX		
DP_LN0_TXN	AH35	DP_LN0_TXN	AIO				AVDD12_DPTX		
DP_LN1_TXP	AJ36	DP_LN1_TXP	AIO				AVDD12_DPTX		
DP_LN1_TXN	AJ37	DP_LN1_TXN	AIO				AVDD12_DPTX		
DP_LN2_TXP	AJ34	DP_LN2_TXP	AIO				AVDD12_DPTX		
DP_LN2_TXN	AK34	DP_LN2_TXN	AIO				AVDD12_DPTX		
DP_LN3_TXP	AL32	DP_LN3_TXP	AIO				AVDD12_DPTX		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
DP_LN3_TXN	AL33	DP_LN3_TXN	AIO				AVDD12_DPTX		
DPAUXP	AG30	DPAUXP	AIO				AVDD12_DPTX		
DPAUXN	AG31	DPAUXN	AIO				AVDD12_DPTX		
EDP_LN0_TXN	AD33	EDP_LN0_TXN	AIO				AVDD12_EDPTX		
EDP_LN0_TXP	AD34	EDP_LN0_TXP	AIO				AVDD12_EDPTX		
EDP_LN1_TXP	AD36	EDP_LN1_TXP	AIO				AVDD12_EDPTX		
EDP_LN1_TXN	AD37	EDP_LN1_TXN	AIO				AVDD12_EDPTX		
EDP_LN2_TXP	AE35	EDP_LN2_TXP	AIO				AVDD12_EDPTX		
EDP_LN2_TXN	AF36	EDP_LN2_TXN	AIO				AVDD12_EDPTX		
EDP_LN3_TXN	AF33	EDP_LN3_TXN	AIO				AVDD12_EDPTX		
EDP_LN3_TXP	AF34	EDP_LN3_TXP	AIO				AVDD12_EDPTX		
EDPAUXN	AE31	EDPAUXN	AIO				AVDD12_EDPTX		
EDPAUXP	AE32	EDPAUXP	AIO				AVDD12_EDPTX		
USB_DP_P0	AM35	USB_DP_P0	AIO				AVDD33_USB_P0		
USB_DM_P0	AM36	USB_DM_P0	AIO				AVDD33_USB_P0		
USB_DP_P1	AC31	USB_DP_P1	AIO				AVDD33_USB_P1		
USB_DM_P1	AC32	USB_DM_P1	AIO				AVDD33_USB_P1		
USB_DP_P2	AD8	USB_DP_P2	AIO				AVDD33_USB_P2		
USB_DM_P2	AE8	USB_DM_P2	AIO				AVDD33_USB_P2		
USB_DP_P3	AH7	USB_DP_P3	AIO				AVDD33_USB_P3		
USB_DM_P3	AG7	USB_DM_P3	AIO				AVDD33_USB_P3		
UFS_PLL_CKREF	G29	UFS_PLL_CKREF	AIO				AVDD12_UFS		
UFS_REFCK_OUT	F28	UFS_REFCK_OUT	AIO				AVDD12_UFS		
UFS_TX0P	C30	UFS_TX0P	AIO				AVDD12_UFS		
UFS_TX0N	C31	UFS_TX0N	AIO				AVDD12_UFS		
UFS_RX0P	B33	UFS_RX0P	AIO				AVDD12_UFS		
UFS_RX0N	B32	UFS_RX0N	AIO				AVDD12_UFS		
UFS_RST_N	A34	UFS_RST_N	AIO				AVDD12_UFS		
AUXIN0	AP5	AUXIN0	AIO				AVDD18_AUXADC		

Ball Name	Ball Location	Signal Name	Signal Type	Aux. Function	Reset Function	Buffer Type	Power Domain	PU/PD Reset	IO Reset
AUXIN1	AP6	AUXIN1	AIO				AVDD18_AUXADC		
AUXIN2	AN6	AUXIN2	AIO				AVDD18_AUXADC		
AUXIN3	AM6	AUXIN3	AIO				AVDD18_AUXADC		
AUXIN4	AL6	AUXIN4	AIO				AVDD18_AUXADC		
AUXIN5	AK6	AUXIN5	AIO				AVDD18_AUXADC		
REFP	AM7	REFP	AIO				AVDD18_AUXADC		
HDMITX21_CH0_M	AR32	HDMITX21_CH0_M	DIO				AVDD18_HDMITX21		
HDMITX21_CH0_P	AR33	HDMITX21_CH0_P	DIO				AVDD18_HDMITX21		
HDMITX21_CH1_M	AT34	HDMITX21_CH1_M	DIO				AVDD18_HDMITX21		
HDMITX21_CH1_P	AU34	HDMITX21_CH1_P	DIO				AVDD18_HDMITX21		
HDMITX21_CH2_M	AR35	HDMITX21_CH2_M	DIO				AVDD18_HDMITX21		
HDMITX21_CH2_P	AR36	HDMITX21_CH2_P	DIO				AVDD18_HDMITX21		
HDMITX21_CLK_M	AT31	HDMITX21_CLK_M	DIO				AVDD18_HDMITX21		
HDMITX21_CLK_P	AU31	HDMITX21_CLK_P	DIO				AVDD18_HDMITX21		
EARCRX_DM	AP31	EARCRX_DM	AIO				AVDD18_EARCRX		
EARCRX_DP	AN31	EARCRX_DP	AIO				AVDD18_EARCRX		
HDMIRX21_CH0_M	AR26	HDMIRX21_CH0_M	AIO				AVDD33_HDMIRX21		
HDMIRX21_CH0_P	AR27	HDMIRX21_CH0_P	AIO				AVDD33_HDMIRX21		
HDMIRX21_CH1_M	AU28	HDMIRX21_CH1_M	AIO				AVDD33_HDMIRX21		
HDMIRX21_CH1_P	AT28	HDMIRX21_CH1_P	AIO				AVDD33_HDMIRX21		
HDMIRX21_CH2_M	AR29	HDMIRX21_CH2_M	AIO				AVDD33_HDMIRX21		
HDMIRX21_CH2_P	AR30	HDMIRX21_CH2_P	AIO				AVDD33_HDMIRX21		
HDMIRX21_CLK_M	AU25	HDMIRX21_CLK_M	AIO				AVDD33_HDMIRX21		
HDMIRX21_CLK_P	AT25	HDMIRX21_CLK_P	AIO				AVDD33_HDMIRX21		

4.3 Power Rails

Table 4-11 lists the device power rails.

Table 4-11 Power Rails

Ball Name	Ball Location	Type	Description
AVDD08_HDMIRX21	AR24	P	Analog power input 0.8 V for HDMI RX
AVDD12_APPLLGP_APU	AC20	P	Analog power for APPLL APU
AVDD12_APPLLGP1	P22	P	Analog power for APPLL
AVDD12_APPLLGP2	V11	P	Analog power for APPLL
AVDD12_APPLLGP3	AF23	P	Analog power for APPLL
AVDD12_APPLLGP4	AC14	P	Analog power for APPLL
AVDD12_AUXADC	AT5	P	Analog power for AUXADC
AVDD12_CKBUF_UFS	H25	P	Analog power for UFS
AVDD12_CKSQ	U31	P	Analog power for CKSQ
AVDD12_CSI	P30	P	Analog power for CSI
AVDD12_DPTX	AD28	P	Analog power for DPTX
AVDD12_DSI	AK1	P	Analog power for DSI
AVDD12_EARCRX	AN30	P	Analog power input for EARCRX
AVDD12_EDPTX	AH29	P	Analog power for EDPTX
AVDD12_HDMITX21	AJ29	P	Analog power input for HDMITX
AVDD12_PCIE_P1	Y29	P	Analog power for PCIe
AVDD12_PCIEG3	AC29	P	Analog power for PCIe Port 0
AVDD12_SSUSB	AN36	P	Analog power for SSUSB
AVDD12_UFS	H24	P	Analog power for UFS
AVDD12_USB_P0	AP37	P	Analog power input USB Port 0
AVDD12_USB_P1	AH30	P	Analog power input USB Port 1
AVDD12_USB_P2	AD6	P	Analog power input USB Port 2
AVDD12_USB_P3	AE6	P	Analog power input USB Port 3
AVDD12_EMI	L8	P	Analog power input EMI
AVDD18_APPLLGP_APU	AC19	P	Analog power input 1.8 V for APPLL
AVDD18_APPLLGP1	R21	P	Analog power input 1.8 V for APPLL
AVDD18_APPLLGP2	W12	P	Analog power input 1.8 V for APPLL
AVDD18_APPLLGP3	AG23	P	Analog power input 1.8 V for APPLL
AVDD18_APPLLGP4	AD14	P	Analog power input 1.8 V for APPLL
AVDD18_AUXADC	AR5	P	Analog power 1.8 V for AUXADC
AVDD18_CKSQ	U30	P	Analog power input 1.8 V for CKSQ
AVDD18_CSI	N30	P	Analog power 1.8 V for CSI
AVDD18_DPTX	AF29	P	Analog power input 1.8 V for DPTX
AVDD18_EARCRX	AM31	P	Analog power 1.8 V for EARCRX
AVDD18_EDPTX	AF37	P	Analog power input 1.8 V for EDPTX
AVDD18_HDMITX21	AL31	P	Analog power input 1.8 V for HDMITX
AVDD18_HDMIRX21	AK24	P	Analog power input 1.8 V for HDMIRX
AVDD18_PCIE_P1	Y30	P	Analog power 1.8 V for PCIe Port 1

Ball Name	Ball Location	Type	Description
AVDD18_PCIEG3	AB30	P	Analog power 1.8 V for PCIe Port 0
AVDD18_PROC	P28	P	Analog power 1.8 V for A78/A55
AVDD18_SSUSB	AN35	P	Analog power 1.8 V for SSUSB
AVDD18_UFS	H26	P	Analog power 1.8 V for UFS
AVDD18_USB_P0	AN34	P	Analog power 1.8 V for USB Port 0
AVDD18_USB_P1	AD29	P	Analog power 1.8 V for USB Port 1
AVDD18_USB_P2	Y8	P	Analog power 1.8 V for USB Port 2
AVDD18_USB_P3	AA8	P	Analog power 1.8 V for USB Port 3
AVDD18_EMI	J8	P	Analog power input EMI
AVDD33_HDMIRX21	AJ26	P	Analog power input 3.3 V for HDMI RX
AVDD33_USB_P0	AM34	P	Analog power 3.3 V for USB Port 0
AVDD33_USB_P1	AD30	P	Analog power 3.3 V for USB Port 1
AVDD33_USB_P2	AB7	P	Analog power 3.3 V for USB Port 2
AVDD33_USB_P3	AC7	P	Analog power 3.3 V for USB Port 3
AVDD075_EMI0	J10, J12, J14, J15, J17, J18	P	Analog power 0.75 V for EMI
AVDD075_EMI2	L9, N9, R9, U9, V9, Y9	P	Analog power 0.75 V for EMI
AVDDQ_EMI0	K11, K13, K14, K16, K18	P	VDDQ power
AVDDQ_EMI2	M10, P10, T10, U10, V10, W10	P	VDDQ power
DVDD_APU	AD21, AD22, AE21, AE22, AF22, AG21, AG22, AH21	P	Digital power input for APU
DVDD_CORE	J21, K21, L14, L17, L18, L21, L22, L23, L30, M12, M13, M14, M17, M20, M21, M23, N17, N18, P13, P14, P17, P18, P23, R13, R14, R17, R18, R22, R23, T13, T14, T21, T22, U13, U14, U17, U18, U21, U22, V13, V14, V17, V18, V21, V22, W13, W14, W17, W18, W21, W22, Y14, Y17, Y18, Y21, Y22, AA18, AA21, AA25, AA26, AB21, AB22, AB25, AB26, AC25, AC26, AD25, AD26, AE26, AE27, AF25, AF26, AG25, AG26, AJ21, AJ23	P	Digital power input for DSP, GPU, and other SoC resources
DVDD_DLA	AB17, AC17, AC18, AD17, AD18, AE17, AF17, AF18, AG17, AG18, AH17, AH18	P	Digital power input for DLA
DVDD_GPU	Y10, Y13, AA10, AA13, AA14, AB10, AB13, AB14, AC10, AC13, AD10, AD13, AE10, AE13, AE14, AF10, AF13, AF14, AG10, AG13, AG14, AH10, AH13, AH14	P	Digital power input for GPU
DVDD_PROC_B	K25, K26, K27, L25, L26, L27, L28, M25, M26, N25, N26, U25, U26, V25, V26, V27, W24, W25, W26, W27, Y25, Y26, Y27	P	Digital power input for A78 core
DVDD_PROC_L	P25, P26, R25, R26, R27, T24, T25, T26	P	Digital power input for A55 core
DVDD_SRAM_APU	AE18, AF21	P	Digital power input for APU SRAM
DVDD_SRAM_CORE	M18, AA17, AE25, AJ8, M22, N13, T12, T19	P	Digital power input for Core SRAM
DVDD_SRAM_GPU	AC12, AD11	P	Digital power input for GPU SRAM

Ball Name	Ball Location	Type	Description
DVDD_SRAM_PROC_B	N29, U23	P	Digital power input for A78 core SRAM
DVDD_SRAM_PROC_L	R28	P	Digital power input for A55 core SRAM
DVDD18_IOBM	AJ22	P	Digital power input for 1.8 V I/O
DVDD18_IOBR	AJ7	P	Digital power input for 1.8 V I/O
DVDD18_IODGI	AU12	P	Digital power input for 1.8 V I/O
DVDD18_IOEMMC	C37	P	Digital power input for MSDC0
DVDD18_IOT	F33	P	Digital power input for 1.8 V I/O
DVDD18_MSDC1	AU5	P	Digital power input for MSDC1
DVDD18_MSDC2	K31	P	Digital power input for MSDC2
DVDD18_VQPS	AB15	P	eFuse blowing power supply
DVDD28_IODGI	AM12	P	Digital power input for I/O
DVDD28_MSDC1	AU6	P	Digital power input for MSDC1
DVDD28_MSDC2	H36	P	Digital power input for MSDC2
DVSS	A17, A32, B4, B5, B7, B9, B11, B13, B22, B24, B26, B28, B30, B34, C2, C4, C8, C10, C21, C23, C25, C29, C32, C33, C34, D4, D6, D7, D11, D12, D13, D15, D18, D26, D27, D30, D31, E2, E5, E6, E10, E21, E24, E28, E29, E31, F4, F9, F16, F18, F30, G2, G4, G15, G22, G23, G24, G25, G26, H3, H29, H30, H35, H37, J2, J4, J5, J19, J20, J28, J32, J33, K3, K20, K22, K24, K28, K29, K30, K32, L2, L5, L16, L19, L20, L24, L29, L31, L35, M4, M11, M15, M16, M19, M24, M27, M28, M29, M31, M34, N2, N7, N11, N12, N15, N16, N19, N24, N27, N28, P11, P12, P16, P20, P24, P27, P29, P31, P32, P33, P34, P35, R11, R12, R15, R19, R24, R31, T2, T6, T11, T15, T16, T20, T23, T27, T28, T31, U7, U11, U12, U15, U16, U19, U20, U24, U27, U28, U29, U32, U33, U34, U35, V6, V7, V12, V15, V16, V19, V20, V23, V24, V28, V30, V31, V32, V35, W2, W4, W5, W6, W11, W15, W16, W19, W20, W23, W28, W30, W33, W34, Y4, Y6, Y7, Y11, Y12, Y15, Y16, Y19, Y20, Y23, Y24, Y28, Y31, Y32, Y33, Y34, AA2, AA4, AA6, AA7, AA11, AA12, AA15, AA16, AA19, AA20, AA23, AA24, AA27, AA28, AA29, AA32, AA35, AB3, AB5, AB6, AB11, AB12, AB16, AB23, AB24, AB27, AB31, AB32, AB33, AB34, AC2, AC9, AC11, AC15, AC16, AC24, AC28, AC30, AC34, AC36, AD3, AD5, AD12, AD15, AD16, AD19, AD20, AD23, AD27, AD31, AD32, AD35, AE2, AE4, AE11, AE12, AE16, AE19, AE20, AE23, AE24, AE33, AE34, AF4, AF6, AF7, AF11, AF12, AF15, AF16, AF19, AF20, AF24, AF27, AF31, AF32, AF35, AG2, AG5, AG6, AG9, AG11, AG12,	G	Digital ground

Ball Name	Ball Location	Type	Description
	AG15, AG16, AG19, AG20, AG28, AG33, AG34, AH3, AH4, AH5, AH6, AH9, AH11, AH12, AH15, AH16, AH19, AH20, AH23, AH25, AH26, AH28, AH31, AH34, AJ2, AJ5, AJ6, AJ30, AJ32, AJ33, AJ35, AK2, AK26, AK30, AK32, AK33, AL5, AL34, AL35, AM1, AM5, AM27, AN3, AN7, AN32, AN33, AN37, AP3, AP4, AP7, AP26, AP27, AP28, AP29, AP30, AP32, AP33, AP35, AP36, AR6, AR7, AR25, AR28, AR31, AR34, AR37, AT2, AT3, AT36, AU36		

4.4 Reserved and Unused Pin Handling Recommendations

MT8395 Baseband Design Notice provides specific pin handling recommendations for the case that the pins are not used.

5 Electrical Characteristics

Stresses above the values listed in Table 5-1 may cause permanent damage to the device. The recommended minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies based on characterization results. Exposure to absolute maximum rating conditions may affect device reliability.

The operating conditions in Table 5-2 must not be exceeded in order to ensure correct operation and reliability of the device. All parameters specified in this document refer to these operating conditions, unless noted otherwise.

5.1 Absolute Maximum Ratings

Table 5-1 represents the absolute maximum ratings of the device power pins.

Table 5-1 Absolute Maximum Ratings

Parameter	Conditions	Max	Unit
Digital power input for A78 core	DVDD_PROC_B, DVDD_SRAM_PROC_B	1.08	V
Digital power input for A55 core	DVDD_PROC_L, DVDD_SRAM_PROC_L	1.08	V
Digital power input for core	DVDD_CORE	0.825	V
Digital power input for GPU	DVDD_GPU	0.8	V
Analog power input	AVDD12_APPLLGP1, AVDD12_APPLLGP2, AVDD12_ APPLLGP3, AVDD12_APPLLGP4, AVDD12_APPLLGP_APU, AVDD12_USB_P0, AVDD12_USB_P1, AVDD12_USB_P2, AVDD12_USB_P3, AVDD12_SSUSB, AVDD12_UFS, AVDD12_CKBUF_UFS, AVDD12_DPTX, AVDD12_EDPTX, AVDD12_HDMITX21, AVDD12_EARCRX, AVDD12_AUXADC, AVDD12_DSI, AVDD12_CSI, AVDD12_PCIE_P1, AVDD12_PCIEG3, AVDD12_CKSQ	1.32	V
	AVDD18_APPLLGP1, AVDD18_APPLLGP2, AVDD18_ APPLLGP3, AVDD18_APPLLGP4, AVDD18_APPLLGP_APU, AVDD18_USB_P0, AVDD18_USB_P1, AVDD18_USB_P2, AVDD18_USB_P3, AVDD18_SSUSB, AVDD18_UFS, AVDD18_DPTX, AVDD18_EDPTX, AVDD18_HDMITX21, AVDD18_HDMIRX21, AVDD18_EARCRX, AVDD18_AUXADC, AVDD18_CSI, AVDD18_PCIE_P1, AVDD18_PCIEG3, AVDD18_CKSQ, AVDD18_PROC, AVDD18_EMI	1.98	V
	AVDD08_HDMIRX21	0.84	V
	AVDD33_USB_P0, AVDD33_USB_P1, AVDD33_USB_P2, AVDD33_USB_P3	3.22	V
	AVDD33_HDMIRX21	3.465	V
	AVDDQ_EMI0, AVDDQ_EMI2	0.65	V
	AVDD075_EMI0, AVDD075_EMI2	0.825	V

Parameter	Conditions	Max	Unit
	AVDD12_EMI	1.26	V
Digital power input	DVDD_SRAM_GPU	0.8	V
	DVDD_SRAM_APU	0.88	V
	DVDD_APU	0.84	V
	DVDD18_IOEMMC, DVDD18_MSDC1, DVDD18_MSDC2, DVDD18_IODGI	1.95	V
	DVDD28_IODGI, DVDD28_MSDC1, DVDD28_MSDC2	3.15	V
	DVDD18_IOT, DVDD18_IOBM, DVDD18_IOBR	1.98	V
	DVDD18_VQPS	1.98	V
	DVDD_DLA	0.866	V
	DVDD_SRAM_CORE	0.787	V
Junction temperature	-	125	°C

5.1.1 Storage Conditions

Table 5-2 defines storage conditions specifics.

Table 5-2 Storage Conditions

Parameter		Min	Max	Unit
Shelf life in sealed bag	40 °C/90% RH		24	months
After bag opened(1)				
Mounted	30 °C/60% RH		168	h
Stored			20	% RH
Baking				
Low temperature device containers	40 °C +5 °C/-0 °C and < 5% RH	192		h
High temperature device containers	125 °C +5 °C/-0 °C	24		h

(1) For devices subjected to infrared reflow, vapor-phase reflow, or equivalent processing.

5.2 Recommended Operating Conditions

Table 5-3 presents the recommended operating conditions of the device power pins.

Table 5-3 Recommended Operating Conditions

Pin Name	Description	Min	Typ	Max	Unit
AVDD12_APPLLGP1	Analog power for APPLLGP1	1.14	1.2	1.26	V
AVDD12_APPLLGP2	Analog power for APPLLGP2	1.14	1.2	1.26	V
AVDD12_APPLLGP3	Analog power for APPLLGP3	1.14	1.2	1.26	V
AVDD12_APPLLGP4	Analog power for APPLLGP4	1.14	1.2	1.26	V
AVDD12_APPLLGP_APU	Analog power for APPLLGP_APU	1.14	1.2	1.26	V
AVDD12_USB_P0	Analog power for USB Port 0	1.14	1.2	1.26	V
AVDD12_USB_P1	Analog power for USB Port 1	1.14	1.2	1.26	V

Pin Name	Description	Min	Typ	Max	Unit
AVDD12_USB_P2	Analog power for USB Port 2	1.14	1.2	1.26	V
AVDD12_USB_P3	Analog power for USB Port 3	1.14	1.2	1.26	V
AVDD12_SSUSB	Analog power for SSUSB	1.14	1.2	1.26	V
AVDD12_UFS	Analog power for UFS	1.14	1.2	1.26	V
AVDD12_CKBUF_UFS	Analog power for UFS	1.14	1.2	1.26	V
AVDD12_DPTX	Analog power for DPTX	1.14	1.2	1.26	V
AVDD12_EDPTX	Analog power for EDPTX	1.14	1.2	1.26	V
AVDD12_HDMITX21	Analog power for HDMITX	1.14	1.2	1.26	V
AVDD12_EARCRX	Analog power	1.14	1.2	1.26	V
AVDD12_AUXADC	Analog power for AUXADC	1.14	1.2	1.26	V
AVDD12_DSI	Analog power for DSI	1.14	1.2	1.26	V
AVDD12_CSI	Analog power for CSI	1.14	1.2	1.26	V
AVDD12_PCIE_P1	Analog power for PCIe Port 1	1.14	1.2	1.26	V
AVDD12_PCIEG3	Analog power for PCIe Port 0	1.14	1.2	1.26	V
AVDD12_CKSQ	Analog power for CKSQ	1.14	1.2	1.26	V
AVDD18_APPLLGP1	Analog power for APPLLGP1	1.71	1.8	1.89	V
AVDD18_APPLLGP2	Analog power for APPLLGP2	1.71	1.8	1.89	V
AVDD18_APPLLGP3	Analog power for APPLLGP3	1.71	1.8	1.89	V
AVDD18_APPLLGP4	Analog power for APPLLGP4	1.71	1.8	1.89	V
AVDD18_APPLLGP_APU	Analog power for APPLLGP_APU	1.71	1.8	1.89	V
AVDD18_USB_P0	Analog power for USB Port 0	1.71	1.8	1.89	V
AVDD18_USB_P1	Analog power for USB Port 1	1.71	1.8	1.89	V
AVDD18_USB_P2	Analog power for USB Port 2	1.71	1.8	1.89	V
AVDD18_USB_P3	Analog power for USB Port 3	1.71	1.8	1.89	V
AVDD18_SSUSB	Analog power for SSUSB	1.71	1.8	1.89	V
AVDD18_UFS	Analog power for UFS	1.71	1.8	1.89	V
AVDD18_DPTX	Analog power for DPTX	1.71	1.8	1.89	V
AVDD18_EDPTX	Analog power for EDPTX	1.71	1.8	1.89	V
AVDD18_HDMITX21	Analog power for HDMITX	1.71	1.8	1.89	V
AVDD18_HDMIRX21	Analog power for HDMIRX	1.71	1.8	1.89	V
AVDD18_EARCRX	Analog power	1.71	1.8	1.89	V
AVDD18_AUXADC	Analog power for AUXADC	1.71	1.8	1.89	V
AVDD18_CSI	Analog power for CSI	1.71	1.8	1.89	V
AVDD18_PCIE_P1	Analog power for PCIe Port 1	1.71	1.8	1.89	V
AVDD18_PCIEG3	Analog power for PCIe Port 0	1.71	1.8	1.89	V
AVDD18_CKSQ	Analog power for CKSQ	1.71	1.8	1.89	V
AVDD18_PROC	Analog power for A78/A55	1.71	1.8	1.89	V
AVDD33_USB_P0	Analog power for USB Port 0	2.92	3.07	3.22	V
AVDD33_USB_P1	Analog power for USB Port 1	2.92	3.07	3.22	V
AVDD33_USB_P2	Analog power for USB Port 2	2.92	3.07	3.22	V
AVDD33_USB_P3	Analog power for USB Port 3	2.92	3.07	3.22	V
AVDD33_HDMIRX21	Analog power for HDMIRX	3.135	3.3	3.465	V

Pin Name	Description	Min	Typ	Max	Unit
AVDD08_HDMIRX21	Analog power for HDMIRX	0.76	0.8	0.84	V
AVDDQ_EMI0	Analog power for EMI0	0.57	0.6	0.65	V
AVDDQ_EMI2	Analog power for EMI2	0.57	0.6	0.65	V
AVDD075_EMI0	Analog power for EMI0	0.675	0.75	0.825	V
AVDD075_EMI2	Analog power for EMI2	0.675	0.75	0.825	V
AVDD12_EMI	Analog power for LPDDR4X	1.14	1.2	1.26	V
AVDD18_EMI	Analog power for LPDDR4X	1.62	1.8	1.98	V
DVDD18_IOT DVDD18_IOBM DVDD18_IOBR	Digital power input for 1.8 V I/O	1.71	1.8	1.89	V
DVDD18_IOEMMC	Digital power input for MSDC0	1.71	1.8	1.89	V
DVDD18_MSDC1	Digital power input for MSDC1	1.71	1.8	1.89	V
DVDD28_MSDC1	Digital power input for MSDC1	2.7	3	3.15	V
DVDD18_MSDC2	Digital power input for MSDC2	1.71	1.8	1.89	V
DVDD28_MSDC2	Digital power input for MSDC2	2.7	3	3.15	V
DVDD18_VQPS	Digital power input for VQPS	1.62	1.8	1.98	V
DVDD18_IODGI	Digital power input for DGI	1.71	1.8	1.89	V
DVDD28_IODGI	Digital power input for DGI	2.7	3	3.15	V
DVDD_CORE	Digital power input for core	0.52	0.75	0.787	V
DVDD_GPU	Digital power input for GPU	0.546	0.75	0.787	V
DVDD_PROC_B	Digital power input for A78 core	0.617	1	1.08	V
DVDD_PROC_L	Digital power input for A55 core	0.617	1	1.08	V
DVDD_APU	Digital power input for APU	0.546	0.8	0.84	V
DVDD_DLA	Digital power input for DLA	0.546	0.775	0.814	V
DVDD_SRAM_CORE	Digital power input for core SRAM	0.712 0.52(1)	0.75	0.787	V
DVDD_SRAM_APU	Digital power input for APU SRAM	0.712 0.52(1)	0.8	0.88	V
DVDD_SRAM_GPU	Digital power input for GPU SRAM	0.712 0.52(1)	0.75	0.787	V
DVDD_SRAM_PROC_B	Digital power input for A78 core SRAM	0.712 0.57(1)	1	1.08	V
DVDD_SRAM_PROC_L	Digital power input for A55 core SRAM	0.712 0.57(1)	1	1.08	V

(1) These values are for SRAM retention only, not for operation.

5.3 DC Electrical Specifications

This section provides DC electrical characteristics per buffer type.

5.3.1 RTCIO DC Specifications

Table 5-4 shows RTC DC buffer (RTCIO) electrical characteristics.

Table 5-4 RTCIO DC Specifications

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.65 × VDDIO(1)		VDDIO(1) + 0.3	V
V_{IL}	Input logic low voltage	-0.3		0.35 × VDDIO(1)	V
F_{RTC}	Input clock frequency		32		kHz
DC_{RTC}	Input signal duty cycle	45	50	55	%
OUTPUT					
V_{OH}	DC output logic high voltage	0.75 × VDDIO(1)			V
V_{OL}	DC output logic low voltage			0.25 × VDDIO(1)	V

(1) VDDIO in this table stands for corresponding power supply (i.e DVDD18_IOBM). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.2 SPII2SIO DC Specifications

Table 5-5 shows SPI, I2S DC buffer (SPII2SIO) electrical characteristics.

Table 5-5 SPII2SIO DC Specifications

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.65 × VDDIO(1)		VDDIO(1) + 0.3	V
V_{IL}	Input logic low voltage	-0.3		0.35 × VDDIO(1)	V
OUTPUT					
V_{OH}	DC output logic high voltage	0.75 × VDDIO(1)			V
V_{OL}	DC Output logic low voltage			0.25 × VDDIO(1)	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD18_IOBM). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.3 I2CI3CIO DC Specifications

Table 5-6 shows I2C/I3C DC buffer (I2CI3CIO) electrical characteristics.

Table 5-6 I2CI3CIO DC Specifications

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.65 × VDDIO(1)		VDDIO(1) + 0.3	V
V_{IL}	Input logic low voltage	-0.3		0.35 × VDDIO(1)	V
OUTPUT					

Parameters		Min	Typ	Max	Unit
V_{OL}	DC output logic low voltage			0.2 × VDDIO(1)	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD18_IOT). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.4 I2C3IO DC Specifications

Table 5-7 shows I2C3 DC buffer (I2C3IO) specifications.

Table 5-7 I2C3IO DC Specifications

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.65 × VDDIO(1)		VDDIO(1) + 0.3	V
V_{IL}	Input logic low voltage	-0.3		0.35 × VDDIO(1)	V
OUTPUT					
V_{OL}	DC output logic low voltage			0.2 × VDDIO(1)	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD18_IOT). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.5 MSDC0IO DC Specifications

Table 5-8 shows MSDC0 DC buffer (MSDC0IO) specifications.

Table 5-8 MSDC0IO DC Specifications

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.65 × VDDIO(1)		VDDIO(1) + 0.3	V
V_{IL}	Input logic low voltage	-0.3		0.35 × VDDIO(1)	V
OUTPUT					
V_{OH}	DC output logic high voltage	1.4			V
V_{OL}	DC output logic low voltage			0.45	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD18_IOEMMC). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.6 MSDC1IO DC Specifications

Table 5-9 shows MSDC1 DC buffer (MSDC1IO) specifications.

Table 5-9 MSDC1IO DC Specifications (2.8 V/3.0 V)

Parameters	Min	Typ	Max	Unit
Operating voltage = 2.8 V/3.0 V				
INPUT				

Parameters		Min	Typ	Max	Unit
V_{IH}	Input logic high voltage	0.75 × VDDIO[(1)]		VDDIO[(1)] + 0.15	V
V_{IL}	Input logic low voltage	-0.3		0.25 × VDDIO[(1)]	V
OUTPUT					
V_{OH}	DC output logic high voltage	0.625 × VDDIO[(1)]		VDDIO[(1)] + 0.15	V
V_{OL}	DC output logic low voltage	-0.3		0.125 × VDDIO[(1)]	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD28_MSDC1). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

Table 5-10 MSDC1IO DC Specifications (1.8 V)

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.7 × VDDIO[(1)]		VDDIO[(1)] + 0.15	V
V_{IL}	Input logic low voltage	-0.3		0.3 × VDDIO[(1)]	V
OUTPUT					
V_{OH}	DC output logic high voltage	1.4		VDDIO[(1)] + 0.15	V
V_{OL}	DC output logic low voltage	-0.3		0.45	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD18_MSDC1). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.7 MSDC2IO DC Specifications

Table 5-11 MSDC2IO DC Specifications (2.8 V/3.0 V)

Parameters		Min	Typ	Max	Unit
Operating voltage = 2.8 V/3.0 V					
INPUT					
V_{IH}	Input logic high voltage	0.75 × VDDIO[(1)]		VDDIO[(1)] + 0.15	V
V_{IL}	Input logic low voltage	-0.3		0.25 × VDDIO[(1)]	V
OUTPUT					
V_{OH}	DC output logic high voltage	0.625 × VDDIO[(1)]		VDDIO[(1)] + 0.15	V
V_{OL}	DC output logic low voltage	-0.3		0.125 × VDDIO[(1)]	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD28_MSDC1). For more information on the power supply name on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

Table 5-12 MSDC2IO DC Specifications (1.8 V)

Parameters		Min	Typ	Max	Unit
Operating voltage = 1.8 V					
INPUT					
V_{IH}	Input logic high voltage	0.7 × VDDIO[(1)]		VDDIO[(1)] + 0.15	V
V_{IL}	Input logic low voltage	-0.3		0.3 × VDDIO[(1)]	V
OUTPUT					
V_{OH}	DC output logic high voltage	1.4		VDDIO[(1)] + 0.15	V
V_{OL}	DC output logic low voltage	-0.3		0.45	V

(1) VDDIO in this table stands for corresponding power supply (i.e. DVDD18_MSDC1). For more information on the power supply name

on the corresponding ball, see Table 4-10 Pin Characteristics, *Power Domain* column.

5.3.8 DDRIO DC Specifications

The EMI LPDDR4X electrical characteristics are compliant with JEDEC Standard—JESD209-4B.

5.4 Power Management

5.4.1 Power Sequences

Refer to the respective PMIC (MT6365/MT6360/MT6315) datasheet for detailed timing sequence.

5.5 Reset

5.5.1 Overview

The Top Reset Generation Unit (TOPRGU) is responsible for generating and distributing reset signals to various subsystems to ensure that the system operates in a stable and reliable manner. It consists of a watchdog timer (WDT), which helps protect against potential system crashes by monitoring system operation and resetting it if an unexpected event occurs.

5.5.2 Features

- Hardware reset signals for the whole chip
- Software controllable reset for subsystems
- Watchdog timer timeout reset
- Reset output signals for companion chips

5.5.3 Block Diagram

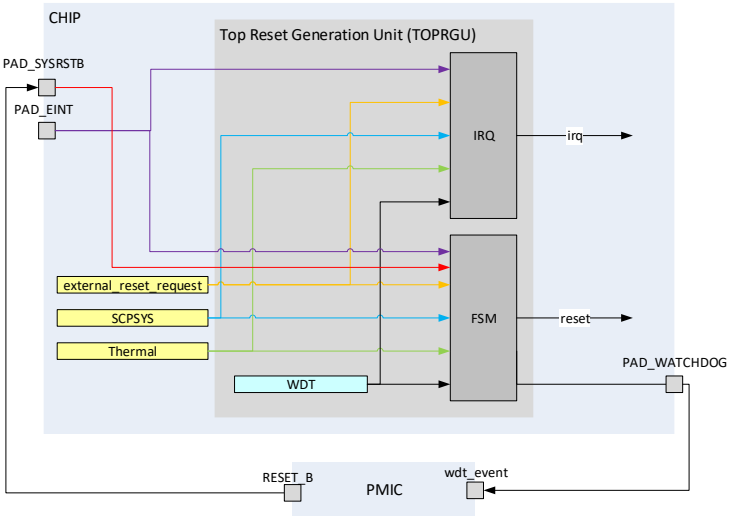


Figure 5-1 Block Diagram of TOPRGU

5.5.4 Function Description

TOPRGU is responsible for generating and distributing reset signals to various subsystems to ensure that the system operates in a stable and reliable manner.

In addition, enhanced features such as watchdog timer, configurable handling mode to reset requests, and dual-mode reset event handling are provided for advanced usage.

- Watchdog timer
  - TOPRGU provides a watchdog timer that generates timeout event when it counts down to 0.
  - Watchdog timer can be used to reset the system or raise interrupt to the system when timeout occurs. For detailed usage, see Sections 5.5.7.1 and 5.5.7.2.
- Configurable handling mode to reset requests
  - A configurable IRQ/reset mode selection is provided for each reset requests to allow advanced event handling. This allows reset requests to be handled as interrupts to fit application needs. See Section 5.5.7.3.
- Dual-mode reset event handling
  - Dual-mode allows watchdog to trigger IRQ first upon reset event, followed by a watchdog reset if the watchdog timer times out again. Detail usage can be found in Section 5.5.7.4.

5.5.5 Reset IO Signal Descriptions

Table 5-13 presents Reset signal description.

Table 5-13 Reset Signal Descriptions

Signal Name	Type	Description	Ball Location
SYSRSTB	DI	System reset input	AL22
PMIC_WATCHDOG	DO	Watchdog reset output	AM22

5.5.6 Theory of Operation

The TOPRGU includes three distinct stages of operation.

1. **The IDLE state:** No reset event has occurred and the watchdog counter is kicked periodically.
2. **The IRQ state:** When a reset event occurs for the first time in the dual-mode, the TOPRGU triggers an interrupt instead of a reset. The watchdog counter automatically restarts and waits for a timeout again. For more details about the dual-mode, refer to Section 5.5.7.4.
3. **The Reset state:** When the watchdog counter times out again, the TOPRGU triggers a watchdog reset to reset the entire chip.

5.5.7 Programming Guide

5.5.7.1 TOPRGU Initialization

The TOPRGU is set to the dual-mode by default; however, there are two ways to change its behavior. Note that the **bold text** in this section indicates the corresponding register name.

1. Trigger PAD_ SYSRSTB to reset the **WDT_MODE** to the default value. Note that this register cannot be reset by the watchdog reset.
2. Program **WDT_MODE**.

5.5.7.2 Program Watchdog Timer Length

The watchdog counter is kicked periodically, with the period referred to as the "Watchdog Timer length". The configuration steps are as follows.

1. Set wdt_en(**WDT_MODE[0]**) to 1'b1.
2. Update **WDT_LENGTH** and trigger **WDT_RESTART**.

5.5.7.3 IRQ Mode

The TOPRGU is set to the dual-mode by default while all reset requests are in the IRQ mode by default. This means that the interrupt is triggered (instead of the watchdog reset) immediately. To trigger the watchdog reset but not the interrupt, the corresponding configuration of each reset request can be changed so that it is configured either as a reset or an IRQ.

5.5.7.4 Dual-Mode Reset

The dual-mode in the TOPRGU means that an IRQ is triggered first when a reset event occurs, followed by a watchdog reset if the watchdog timer times out again.

The watchdog timer auto-restarts after an interrupt is triggered. The AP should clear the **WDT_STA** after receiving the interrupt from the TOPRGU. When the interrupt is issued, the watchdog timer is restarted to {**WDT_LENGTH**, {9{1'b1}}} and the watchdog reset is triggered when the timer expires. {**WDT_LENGTH**, {9{1'b1}}} means that the WDT timeout period is a multiple of 512*T32k = 15.6 ms.

To configure the dual-mode, perform the following steps:

1. Set wdt_en (**WDT_MODE[0]**) = 1'b1.
2. Set dual_mode (**WDT_MODE[6]**) = 1'b1.

5.5.8 Register Definition

Refer to "*MT8395 Register Map*" for detailed register descriptions.

5.6 DSI Specifications

Table 5-14 presents MIPI D-PHY TX electrical characteristics.

Table 5-14 DSI D-PHY TX Electrical Characteristics

Description	Min	Typ	Max	Unit
High-Speed data rate	125		1200	Mbps
High-Speed common mode voltage	150	200	250	mV
High-Speed differential output voltage	140	200	270	mV
High-Speed single ended output high voltage			360	mV
High-Speed single ended output impedance	40	50	62.5	Ω
High-Speed 20%-80% rise time and fall time			0.3(1)	UI
			0.35(2)	UI
	100(3)			ps
Low-Power output high level	0.95	1.2	1.3	V
Low-Power output low level	-50		50	mV
Low-Power output impedance	110			Ω
Low-Power 15%-85% rise time and fall time			25	ns

(1) Applicable when supporting maximum HS bit rates ≤ 1 Gbps (UI ≥ 1 ns)

(2) Applicable when supporting maximum HS bit rates > 1 Gbps (UI ≤ 1 ns) but ≤ 1.5 Gbps

(3) Applicable when supporting maximum HS bit rates > 1.5 Gbps

Table 5-15 presents MIPI C-PHY TX electrical characteristics.

Table 5-15 DSI C-PHY TX Electrical Characteristics

Description	Min	Typ	Max	Unit
High-Speed data rate	125		1100	Msps
High-Speed common mode voltage	175	225 - 250	310	mV
High-Speed differential output voltage of strong one			300	mV
High-Speed differential output voltage of weak one	97			mV
High-Speed single ended output high voltage			425	mV
High-Speed single ended output impedance	40	50	60	Ω
High-Speed rise time and fall time from -58 mV to 58 mV			0.4	UI
Low-Power output high level	0.95	1.2	1.3	V
Low-Power output low level	-50		50	mV
Low-Power output impedance	110			Ω
Low-Power 15%-85% rise time and fall time			25	ns

5.7 CSI-2 Specifications

Table 5-16 presents MIPI D-PHY RX electrical characteristics.

Table 5-16 CSI D-PHY RX Electrical Characteristics

Description	Min	Typ	Max	Unit
High-Speed data rate	80		2500	Mbps
High-Speed common point voltage	70		330	mV

Description	Min	Typ	Max	Unit
High-Speed differential input high voltage			40	mV
High-Speed differential input low voltage	-40			mV
High-Speed single ended input high voltage			460	mV
High-Speed single ended input low voltage	-40			mV
High-Speed single ended input impedance	80	100	125	Ω
Low-Power logic 1 input voltage	740			mV
Low-Power logic 0 input voltage			550	mV
Low-Power input hysteresis	25			mV
Minimum pulse width response	20			ns

6 Clock Characteristics

The device has two external input clocks—low frequency (RTC32K_CK) and high frequency (X26M_IN). Figure 6-1 shows the external clock sources and clock outputs.

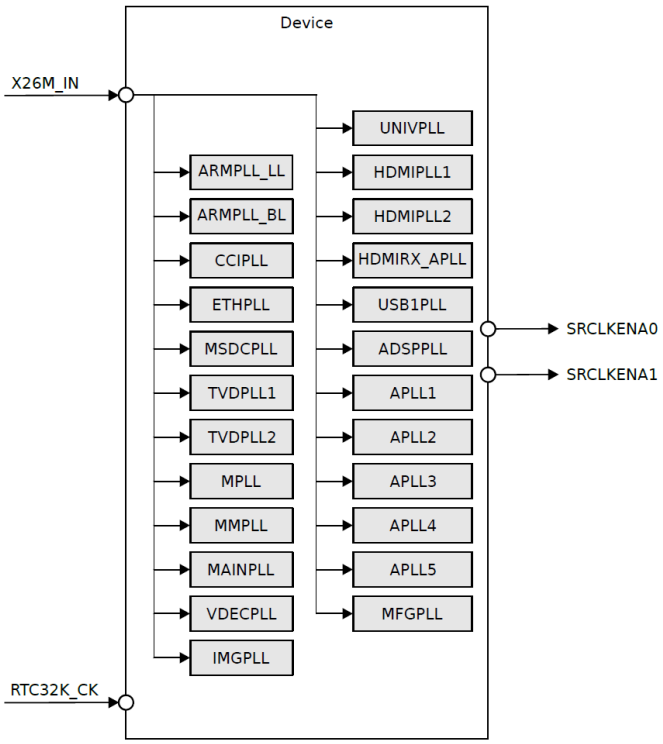


Figure 6-1 Device Clock Diagram

6.1 Maximum Performance Ratings

Table 6-1 presents the maximum core and peripheral performance limitations and correlations.

Table 6-1 Maximum Performance Ratings

Module		Max	Unit
Quad-core Arm Cortex-A78	A78	2200	MHz
Quad-core Arm Cortex-A55	A55	2000	MHz
Graphics Accelerator	GPU	880	MHz
HiFi 4 DSP	DSP	720	MHz
AI Processor Unit	APU	832	MHz
System Companion Processor	SCP	416	MHz
External Memory Interface	LPDDR4X	4266	MHz
Memory Card Controller	SD Card	100	MBps
	eMMC	400	MBps
	SDIO	100	MBps
SPI NAND Flash Interface	SNFI (output)	104	MHz

Module		Max	Unit
Serial NOR Flash Interface	SNOR	25	MHz
Universal Flash Storage	UFS	5.8	Gbps
Digital Display Parallel Interface	DPI	148.5	MHz
High-Definition Multimedia Interface Transmitter	HDMITX	594	MHz
DisplayPort	DPTX	8.1	Gbps/lane
Embedded DisplayPort	EDPTX	5.4	Gbps/lane
Display Serial Interface	DSI D-PHY	1.2	Gbps/lane
	DSI C-PHY	1.1	Gsps/trio
Image Signal Processor	ISP	48	MPix@30fps
Camera Serial Interface 2	CSI D-PHY	2.5	Gbps/lane
	CSI C-PHY	4.5	Gsps/trio
High-Definition Multimedia Interface Receiver	HDMIRX	594	MHz
Video Encoder	VENC	624	MHz
Video Decoder	VDEC	312	MHz
Inter-IC Sound	I2S master mode (sampling frequency)	384	kHz
	I2S slave mode (sampling frequency)	48	kHz
Programmable Command Master Interface	PCM (sampling frequency)	48	kHz
Pulse Density Modulation	PDM	3.25	MHz
Time Division Multiplexed Interface	TDM (sampling frequency)	192	kHz
Digital Interface	SPDIF	192	kHz
Inter-Integrated Circuit	I2C mode	3.4	Mbps
	I3C mode	12.5	Mbps
Universal Asynchronous Receiver/Transmitter	UART	961,200	bps
Serial Peripheral Interface	SPI (master)	52	MHz
Universal Serial Bus	USB SuperSpeed	5	Gbps
	USB High-Speed	480	Mbps
	USB Full-Speed	12	Mbps
	USB Low-Speed	1.5	Mbps
Ethernet Network Interface Controller	MII	25	MHz
	RMII	50	MHz
	RGMII	125	MHz
Peripheral Component Interconnect Express	PCIe	8.0	GT/s
Pulse Width Modulation	PWM	39	MHz
Auxiliary ADC	AUXADC (clock rate)	3.25	MHz

6.2 PLL Specifications

Table 6-2 shows ARMPLL_LL specifications.

Table 6-2 ARMPLL_LL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		2000		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-3 shows ARMPLL_BL specifications.

Table 6-3 ARMPLL_BL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		2252.25		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-4 shows CCIPLL specifications.

Table 6-4 CCIPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		1800		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-5 shows ETHPLL specifications.

Table 6-5 ETHPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		500		MHz
t_{SET}	Settling time		20		µs

Parameter		Min	Typ	Max	Unit
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-6 shows MSDCPLL specifications.

Table 6-6 MSDCPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		416.146		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-7 shows TVDPLL1 specifications.

Table 6-7 TVDPLL1 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		594.177		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-8 shows TVDPLL2 specifications.

Table 6-8 TVDPLL2 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		594.177		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V

Parameter		Min	Typ	Max	Unit
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-9 shows MPLL specifications.

Table 6-9 MPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		208.03		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-10 shows MMPLL specifications.

Table 6-10 MMPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		2750.048		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-11 shows MAINPLL specifications.

Table 6-11 MAINPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		2184.359		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-12 shows VDECPLL specifications.

Table 6-12 VDECPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		680.04		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-13 shows IMGPLL specifications.

Table 6-13 IMGPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		650.026		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-14 shows UNIVPLL specifications.

Table 6-14 UNIVPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		2496.004		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-15 shows HDMIPLL1 specifications.

Table 6-15 HDMIPLL1 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		884.016		MHz
t_{SET}	Settling time		20		µs

Parameter		Min	Typ	Max	Unit
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-16 shows HDMIPLL2 specifications.

Table 6-16 HDMIPLL2 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		600.024		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-17 shows HDMIRX_APLL specifications.

Table 6-17 HDMIRX_APLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		294.915		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-18 shows USB1PLL specifications.

Table 6-18 USB1PLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		192.001		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V

Parameter		Min	Typ	Max	Unit
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-19 shows ADSPPLL specifications.

Table 6-19 ADSPPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		720		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-20 shows APLL1 specifications.

Table 6-20 APLL1 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		196.001		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-21 shows APLL2 specifications.

Table 6-21 APLL2 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		180.001		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-22 shows APLL3 specifications.

Table 6-22 APLL3 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		196.001		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-23 shows APLL4 specifications.

Table 6-23 APLL4 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		196.001		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-24 shows APLL5 specifications.

Table 6-24 APLL5 Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		196.001		MHz
t_{SET}	Settling time		20		µs
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

Table 6-25 shows MFGPLL specifications.

Table 6-25 MFGPLL Specifications

Parameter		Min	Typ	Max	Unit
f_{IN}	Input clock frequency		26		MHz
f_{OUT}	Output clock frequency		240		MHz
t_{SET}	Settling time		20		µs

Parameter		Min	Typ	Max	Unit
$f_{OUT(D)}$	Output clock duty cycle	47	50	53	%
$t_{J(CLK)}$	Output clock jitter (period jitter)			60	ps
DVDD	Digital power supply	0.54	0.8	0.88	V
AVDD18	Analog power supply	1.71	1.8	1.89	V
AVDD12	Analog power supply	1.14	1.2	1.26	V

6.3 Clock Squarer

The Clock Squarer (CKSQ) is designed to receive clock signal from the X26M_IN input and distribute it to the chip internally.

Table 6-26 shows the CKSQ specifications.

Table 6-26 CKSQ Specifications

Parameter		Min	Typ	Max	Unit
V_{IN}	Input signal amplitude	1000	1200	1250	mVpp
D_{cycIN}	Input signal duty cycle		50		%
D_{cycOUT}	Output signal duty cycle	D_{cycIN}-5		D_{cycIN}+5	%
	Maximum positive overshoot			1.3	V
	Minimum negative overshoot	-0.1			V

6.4 Clock Signal Descriptions

Table 6-27 presents clock signal descriptions.

Table 6-27 Clock Signal Descriptions

Signal Name	Type	Description	Ball Location
RTC32K_CK	DI	RTC 32 kHz clock input	AR21
X26M_IN	AI	26 MHz clock input	AB28
SRCLKENA0	DO	Output signal; control of PMIC 26 MHz/Buck/LDO normal mode or sleep High: Normal mode Low: Sleep mode or low power mode	AM21
SRCLKENA1	DO	Output signal; control of PMIC 26 MHz/Buck/LDO on or off	AL21

7 Package Information

7.1 Thermal Specifications

7.1.1 Thermal Operating Specifications

Table 7-1, Table 7-2 and Table 7-3 present the thermal resistance characteristics and maximum operating temperatures of the device.

Table 7-1 MT8395AV/ZA Thermal Operating Specifications

Parameter		Value	Unit
θ_{JA}	Package thermal resistances in natural convection	21.5	°C/Watt
θ_{JB}	Package thermal resistance (junction-to-board)	3.5	°C/Watt
θ_{JC}	Package thermal resistance (junction-to-case)	2.36	°C/Watt
T_J	Maximum operating junction temperature	95	°C

Table 7-2 MT8395IV/ZA Thermal Operating Specifications

Parameter		Value	Unit
θ_{JA}	Package thermal resistances in natural convection	21.5	°C/Watt
θ_{JB}	Package thermal resistance (junction-to-board)	3.5	°C/Watt
θ_{JC}	Package thermal resistance (junction-to-case)	2.36	°C/Watt
T_J	Maximum operating junction temperature	105	°C

Table 7-3 MT8395IV/KZA Thermal Operating Specifications

Parameter		Value	Unit
θ_{JA}	Package thermal resistances in natural convection	21.2	°C/Watt
θ_{JB}	Package thermal resistance (junction-to-board)	3.5	°C/Watt
θ_{JC}	Package thermal resistance (junction-to-case)	0.68	°C/Watt
T_J	Maximum operating junction temperature	105	°C

7.2 Top Marking

Figure 7-1 shows the device top marking definition.

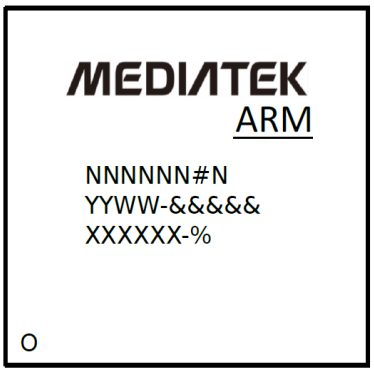


Figure 7-1 Device Top Marking

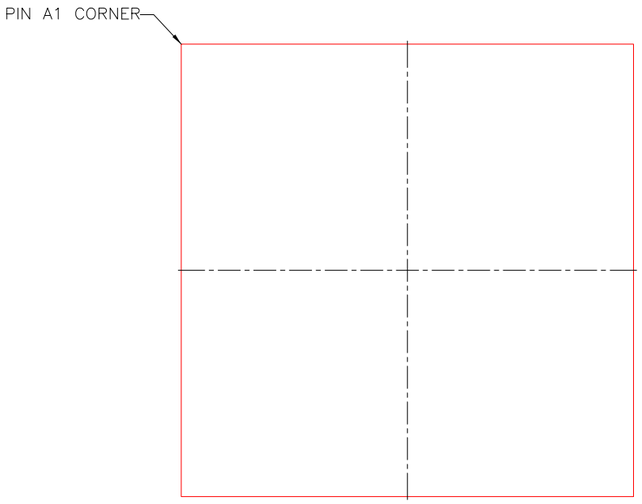
Table 7-4 presents the printed device reference and decoding.

Table 7-4 Printed Device Reference and Decoding

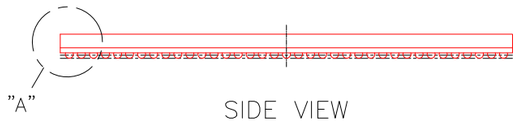
Parameter		Value	Description
NNNNNN#N	Part number	MT8395#V	Multimedia system
#	Function code 1	-	A for consumer-grade I for industrial-grade
YYWW	Date code	-	2-digits year and week code
&&&&&	Random code	-	For internal use only
XXXXXX	Lot number	-	For internal use only
%	Function code 2	-	Blank: PKG θ_{JC}=2.36°C/W K: PKG θ_{JC}=0.68°C/W
O	Pin one designator	-	Pin one location

7.3 Mechanical Drawing

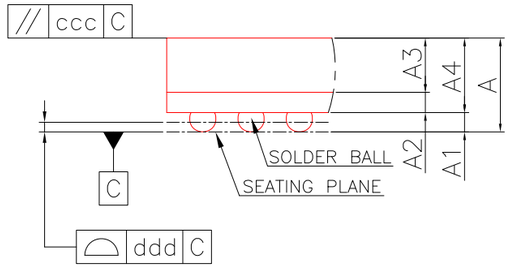
The following figure shows printed device reference diagram (MFC VFBGA 15.0 mm × 15.0 mm, 1046-ball, 0.4 mm pitch package).



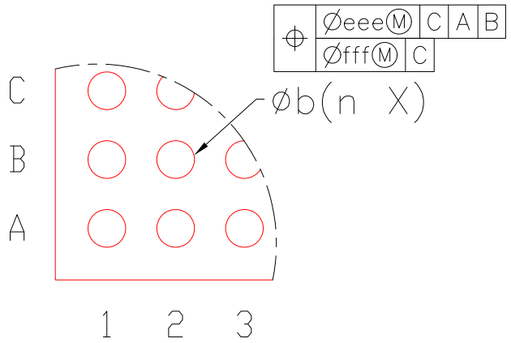
TOP VIEW



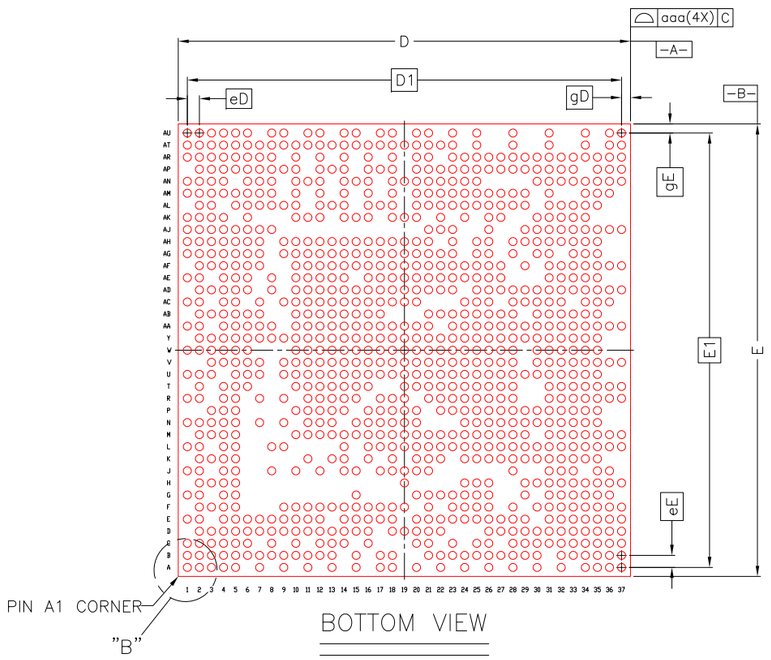
SIDE VIEW



DETAIL : "A"



DETAIL : "B"



BOTTOM VIEW

Item		Symbol	Common Dimensions MIN.	NOM.	MAX.
Package Type			MFC VFBGA		
Body Size	X	D	14.950	15.000	15.050
	Y	E	14.950	15.000	15.050
Ball Pitch	X	eD		0.400	
	Y	eE		0.400	
Mold Thickness		A3		0.450 Ref.	
Substrate Thickness		A2		0.168 Ref.	
Substrate+Mold Thickness		A4	0.568	0.618	0.668
Total Thickness		A	–	–	0.900
Ball Diameter				0.250	
Ball Stand Off		A1	0.140	0.180	0.220
Ball Width		b	0.220	0.270	0.320
Package Edge Tolerance		aaa		0.050	
Mold Flatness		ccc		0.100	
Coplanarity		ddd		0.100	
Ball Offset (Package)		eee		0.150	
Ball Offset (Ball)		fff		0.050	
Ball Count		n		1046	
Edge Ball Center to Center	X	D1		14.400	
	Y	E1		14.400	
Edge Ball Center to Package Edge	X	gD		0.300	
	Y	gE		0.300	

TITLE	PACKAGE OUTLINE MFC VFBGA 1046L 15 X 15 X 0.9 max		
DWG. NO.	REV.	SHEET	UNIT
MT-SP01479	A	1 OF 2	MM

8 Legal and Support Information

8.1 Related Documents and Products

Documents:

- **MMD (MediaTek Module Design)**—Power Delivery Network (PDN) and DRAM design implementation solutions
- **MT8395 Baseband Design Notice**—Application note including schematic examples for peripheral interfaces such as GPIO, MSDC, NAND flash, UFS, LPDDR4X, I2C/I3C, SPI, Display, Camera, USB, DPI, Ethernet, HDMI, PCIe, Audio, and power design implementation recommendations.
- **MT6315 Application Note for MT8395**—MediaTek MT6315 PMIC application note covering functional description and PCB layout guidelines.
- **MT6360 Application Note for MT8395**—MediaTek MT6360 PMIC application note covering functional description and PCB layout guidelines.
- **MT6365 Application Note for MT8395**—MediaTek MT6365 PMIC application note covering functional description and PCB layout guidelines.

Companion chips:

- **MT6315**—Integrated Power Management IC (PMIC)
- **MT6360**—Integrated Power Management IC
- **MT6365**—Integrated Power Management IC

8.2 Trademarks

Arm, Jazelle, Cortex, TrustZone, AMBA, and Thumb are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Mali, CoreSight, Neon, and DynamIQ are trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Vulkan is a registered trademark of the Khronos Group, Inc.

PCIe is a registered trademark of PCI-SIG.

Cadence and Tensilica are registered trademarks of Cadence Design Systems, Inc. HDMI is a trademark of HDMI Licensing, LLC.

I^2C and I^2S are trademarks of NXP Semiconductors.

OpenCL is a trademark of Apple, Inc.

SD is a registered trademark of SD Card Association.

MMC and eMMC are trademarks of MultiMediaCard Association.

MIPI, MIPI I3C, MIPI M-PHY, and UniPro are registered trademarks of MIPI Alliance, Inc. DisplayPort is a trademark of VESA.

Linux is a registered trademark of Linus Torvalds.

Android is a trademark of Google LLC.
Intel is a registered trademark of Intel Corporation or its subsidiaries.

All other product or service names are the property of their respective owners.

Exhibit 1 Terms and Conditions

Your access to and use of this document and the information contained herein (collectively this "Document") is subject to your (including the corporation or other legal entity you represent, collectively "You") acceptance of the terms and conditions set forth below ("T&C"). By using, accessing or downloading this Document, You are accepting the T&C and agree to be bound by the T&C. If You don't agree to the T&C, You may not use this Document and shall immediately destroy any copy thereof.

This Document contains information that is confidential and proprietary to MediaTek Inc. and/or its affiliates (collectively "MediaTek") or its licensors and is provided solely for Your internal use with MediaTek's chipset(s) described in this Document and shall not be used for any other purposes (including but not limited to identifying or providing evidence to support any potential patent infringement claim against MediaTek or any of MediaTek's suppliers and/or direct or indirect customers). Unauthorized use or disclosure of the information contained herein is prohibited. You agree to indemnify MediaTek for any loss or damages suffered by MediaTek for Your unauthorized use or disclosure of this Document, in whole or in part.

MediaTek and its licensors retain titles and all ownership rights in and to this Document and no license (express or implied, by estoppels or otherwise) to any intellectual propriety rights is granted hereunder. This Document is subject to change without further notification. MEDIATEK DOES NOT ASSUME ANY RESPONSIBILITY ARISING OUT OF OR IN CONNECTION WITH ANY USE OF, OR RELIANCE ON, THIS DOCUMENT, AND SPECIFICALLY DISCLAIMS ANY AND ALL LIABILITY, INCLUDING, WITHOUT LIMITATION, CONSEQUENTIAL OR INCIDENTAL DAMAGES.

THIS DOCUMENT AND ANY OTHER MATERIALS OR TECHNICAL SUPPORT PROVIDED BY MEDIATEK IN CONNECTION WITH THIS DOCUMENT, IF ANY, ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE. MEDIATEK SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, COMPLETENESS OR ACCURACY AND ALL WARRANTIES ARISING OUT OF TRADE USAGE OR OUT OF A COURSE OF DEALING OR COURSE OF PERFORMANCE. MEDIATEK SHALL NOT BE RESPONSIBLE FOR ANY MEDIATEK DELIVERABLES MADE TO MEET YOUR SPECIFICATIONS OR TO CONFORM TO A PARTICULAR STANDARD OR OPEN FORUM.

Without limiting the generality of the foregoing, MEDIATEK MAKES NO WARRANTY, REPRESENTATION OR GUARANTEE REGARDING THE SUITABILITY OF ITS PRODUCTS FOR ANY PARTICULAR PURPOSE, NOR DOES MEDIATEK ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT, CIRCUIT OR SOFTWARE. You agree that You are solely responsible for the designing, validating and testing Your product incorporating MediaTek's product and ensure such product meets applicable standards and any safety, security or other requirements.

The above T&C and all acts in connection with the T&C or this Document shall be governed, construed and interpreted in accordance with the laws of Taiwan, without giving effect to the principles of conflicts of law.